ARTIFICIAL INTELLIGENCE RESEARCH BRANCH

1990 Progress Report and Future Plans

Report RIA-90-4-27-1
April, 1990

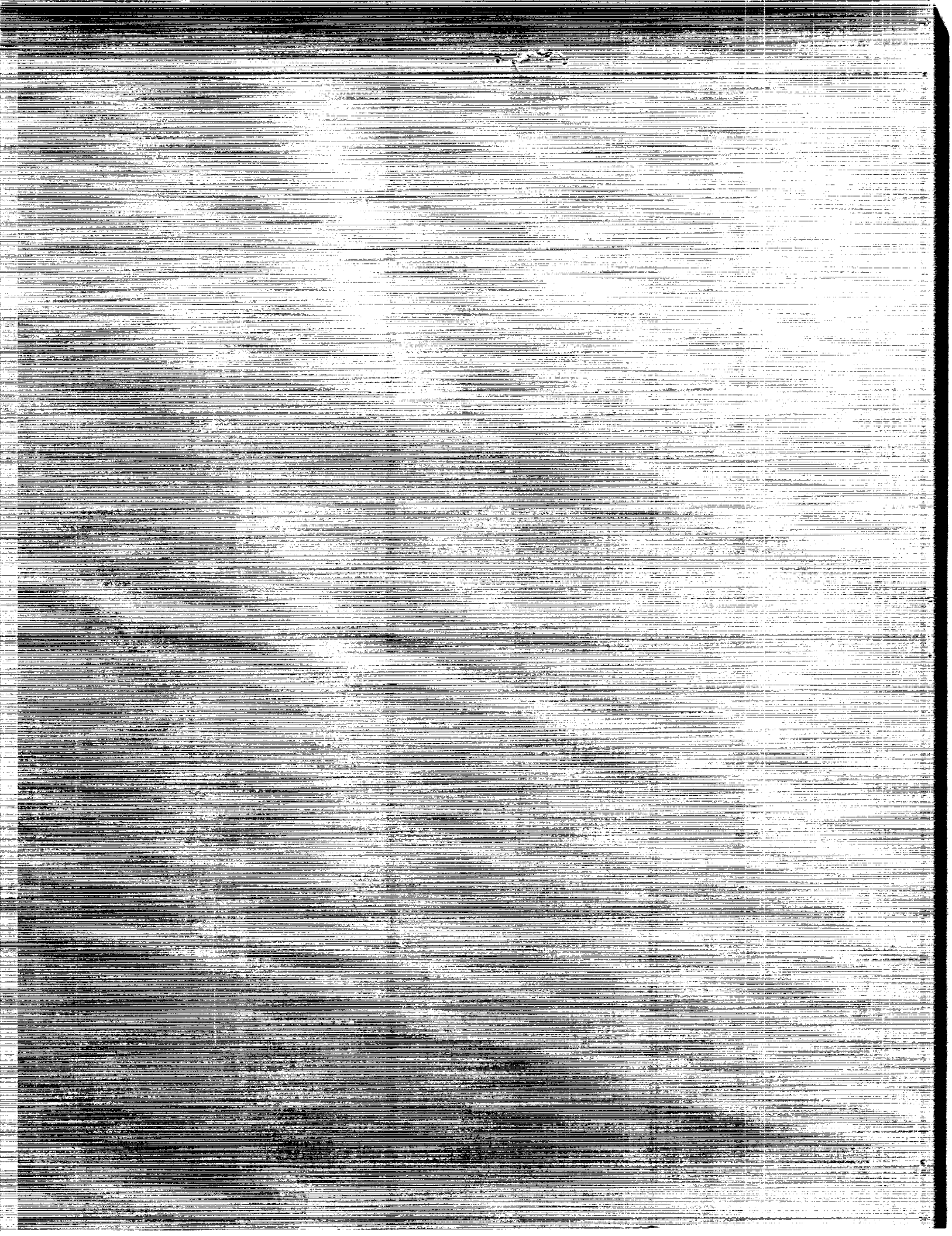

Table of Contents

1. Introduction

This document describes the progress and plans of the Artificial Intelligence Research Branch (RIA) at the NASA Ames Research Center (ARC) in 1990. Activities span a range from basic scientific research through engineering development to fielded NASA applications, particularly those applications that are enabled by basic research carried out in RIA. Work is conducted in-house and through collaborative partners in academia and industry. Our major focus is on a limited number of research themes with a dual commitment to technical excellence and proven applicability to NASA short, medium, and long-term problems. RIA acts as the Agency's lead organization for research aspects of artificial intelligence, working closely with a second research laboratory at the Jet Propulsion Laboratory (JPL) and AI applications groups at all NASA centers.

There are currently forty-five members of the technical staff in RIA, with 18 holding a PhD in computer science or related fields and an additional 20 holding other advanced degrees. RIA staff participate actively in both the artificial intelligence and aerospace professional communities, serving as editors and editorial board members of several major journals, as members of the program committee of the National Conference on Artificial Intelligence, and as members of Technical Committees of the American Institute of Aeronautics and Astronautics (AIAA).

The following is a description of our program according to the NASA requirements we are addressing:

- Intelligent Assistance for Mission Operations: the development of AI-based systems that can act as "intelligence amplifiers" for ground and space-based humans who have the responsibility to conduct mission operations.

- Scientific and Engineering Data Analysis Tools: the development of tools to assist in the analysis of vast amounts of science and engineering data resulting from NASA missions.

- Onboard Systems for Diagnosis, Planning and Intelligent Control: the development of real-time, in-the-loop systems for planning, control, diagnosis and fault correction of current and future spaceborne systems.

- Capture, Integration, and Preservation of Life-Cycle Knowledge: the development of mechanisms for acquiring, combining, maintaining, and utilizing knowledge relating to the devices NASA designs, builds, and operates over long life-spans.

Technically, several major research themes cross all of the functional domains. These are:

- Planning and Scheduling: the synthesis of a set of interacting activities placed in timelines subject to complex resource and domain constraints.

- Machine Learning: techniques for forming theories about natural and man-made phenomena; and for improving the problem-solving performance of computational systems over time.

- Construction, Maintenance, and Utilization of Large-Scale Knowledge Bases: research on knowledge acquisition, knowledge representation, combination of knowledge from many different sources, and multi-purpose utilization of knowledge bases large enough to represent substantial parts of complex NASA devices such as the Hubble Space Telescope and Space Station Freedom.

- Human-Machine Aspects of AI-Based Systems: many of the current and future applications of artificial intelligence in NASA involve a strong interaction with highly trained humans (scientists, engineers, astronauts, etc.). Ensuring the ability to communicate with those humans in an effective manner is vital to eventual user acceptance of all of the other research themes.

Our work is funded by a variety of sources from NASA and other federal agencies. The single largest sponsor of work is the Artificial Intelligence Program of the Information Science and Human Factors Division of the NASA Office of Aeronautics, Exploration, and Technology (OAET). The OAET Exploration Technology Program funds our Planetary Rover Project. (Dr. Mel Montemerlo is the manager of both of these programs). We also receive significant support from the Space Station Freedom Advanced Development Program. (Mr. Gregg Swietek is the responsible program manager.) The Defense Advanced Research Projects Agency's Information Science and Technology Office (DARPA/ISTO) (Major Steve Cross is program manager) and the Air Force Office of Scientific Research (AFOSR) (Dr. Abe Waksman is program manager) co-fund our work in Intelligent Interacting Agents.

2. Intelligent Assistance for Mission Operations

2.1 Automated Scheduling Tools

This research concentrates on solving complex scheduling and resource allocation problems which are prevalent throughout NASA. A typical problem of this sort is one in which a large number of tasks must be assigned start and end times, subject to temporal and resource constraints. Temporal constraints include explicit deadlines as well as ordering constraints (e.g., one task must follow another). Tasks generally require resources to accomplish their goals. Therefore, the assignment of times to the tasks must take into account the availability of the domain resources. A completed schedule is one in which all tasks are assigned times that satisfy the temporal constraints and for which there are sufficient resources.

In general, scheduling is an intractable combinatoric search problem. There are many potential assignments of times and resources which are computationally difficult to coordinate. Additionally, many of the NASA scheduling problems require extensible systems that adapt quickly to changes in the specified problem. Therefore, our program in scheduling concentrates on addressing the dynamic nature of scheduling problems as well as the huge search space normally encountered in such problems.

Constraint-Based Scheduling

Monte Zweben, Megan Eskey, Todd Stock, Will Taylor; RIA
Ellen Drascher; ARC Aerospace Human Factors Division
Bob Gargan, Michael Deale, Michael Pontecorvo, Brian Daun; Lockheed AI Center

In 1990, the constraint-based scheduling project is concentrating on four major activities: the development of a generic scheduling and rescheduling tool; the study of iterative improvement search algorithms for scheduling; the application of machine learning to scheduling; and the application and evaluation of the scheduling tool at the Kennedy Space Center (KSC) for Space Shuttle (STS) processing.

The scheduling tool being developed contains an activity description language, an extensible constraint language, a domain description language, a search control rule system, and an interactive scheduling interface. Users declare the activities they wish to schedule with the activity language, relate these activities to each other and to objects in the domain with constraints, and provide domain scheduling knowledge in the form of search control rules. The system then takes this information and finds times and resource assignments for each task such that all the domain constraints are met. One important point to stress is that the system allows one to express constraints on any kind of time-varying information; that is, it is not restricted to modeling only resource availability over time. Examples of this include device states, switch and valve positions, locations, and sensor values. Further, the activity language allows one to

express the effects that activities have on these "state variables" in addition to the changes tasks make to resource availability. In 1990, we will complete the development of the first version of this tool, continuing our experiments with the KSC cargo processing domain. In 1991, we plan to concentrate on two major topics for tool development. The first is performance; the tool will be optimized for efficiency. The second topic is utility analysis and optimization. In many cases it is essential to develop schedules that "minimize" lateness or work-in-process time. We will extend the search control rule system to support this global optimization criteria.

One of the principal goals of this work is to develop efficient algorithms for rescheduling. In most applications, the *a priori* synthesis of a schedule is important, but equally important is the ability to reactively modify a schedule in reaction to changes that occur during its execution. We have developed rescheduling algorithms that allow users to modify tasks in terms of their start and end times, their constraints, their resource requirements, and their durations.

Our desire for efficient dynamic rescheduling algorithms has resulted in the exploration of iterative improvement scheduling algorithms. These techniques differ from traditional algorithms in that they incrementally repair complete solutions to the scheduling problem rather than systematically extending a partial solution to the problem. Specifically, we have developed a framework called "Constraint-Based Simulated Annealing" which converges to a solution by making local repairs to the violated constraints of some approximately correct schedule. We have developed a number of versions of the Constraint-Based Simulated Annealing algorithm with two major results. First, the new algorithm is at least twice as fast on test problems as conventional scheduling techniques. Second, it is an "anytime" algorithm; that is, at any point the algorithm can be stopped and a solution is returned, with the solution improving the longer the algorithm runs. In 1991, we plan to continue empirical experiments with the iterative improvement algorithms. In addition, we will begin to tackle the extreme combinatoric nature of large scheduling problems by taking advantage of the inherently parallel nature of the Constraint-Based Simulated Annealing algorithm and attempting an implementation on the massively parallel Connection Machine available at ARC.

In 1990 we began experiments with machine learning as a method for improving scheduling systems. In previous research at Carnegie-Mellon University, it was shown that the performance of a scheduler can be greatly improved if the system recognizes resource bottlenecks and then chooses the resources for an activity requiring a congested resource before choosing activity times. Therefore, we believe that a system should learn when to change its search strategy by analyzing its search progress and learning the general conditions under which a resource bottleneck is likely to occur. We are implementing an analytical learning technique called Plausible Explanation-Based Learning (PEBL) that accomplishes this. PEBL extends standard Explanation-Based Learning (EBL) with the addition of an empirical component. This component is necessary because it is not sufficient to conclude that a

chronic resource bottleneck exists from only a single example; multiple examples are needed to gain confidence in the conclusion that a chronic bottleneck exists. When the scheduler reaches a backtracking point, the system tries to explain why it failed. It can do so either by "re-playing" previous explanations or synthesizing new ones. If a previous explanation is used and is successful, then its probability of being accurate is increased. When the confidence in a given explanation reaches some threshold, it is transformed into a search control rule that alters the default search strategy accordingly. New explanations are stored away until their confidences reach the threshold or until they are considered useless. In 1990 we will complete the development and experimentation of this technique. In 1991, we plan to augment our learning techniques to learn constraint orderings and value preferences. We also plan to extend our learning techniques to the iterative improvement search algorithms.

The final activity in this project is the evaluation of the scheduling tool at KSC for STS processing. RIA, the Lockheed AI Center (LAIC), Lockheed Space Operations Center (LSOC), and the KSC Advanced Projects Branch have teamed to augment the existing planning and scheduling tools available at KSC. This project is co-funded by the Advanced Development Program within the Office of Space Flight, and could have a major impact on Shuttle ground operations. We have begun working with KSC schedulers to determine their needs from both a hardware and software perspective. Their current tools are deficient in four major ways. First, they do not support full scheduling and resource allocation. They only provide a scheduling capability comparable to PC project management tools (i.e., PERT/CPM) and have very limited resource leveling capabilities. Second, their tools do not enable the schedulers to represent any constraints other than predecessor and successor relations between tasks. This is a problem because some tasks, such as hazardous tasks, are not causally required to be before or after another task, but instead cannot be accomplished in parallel with other tasks. These tools require schedulers to commit to an arbitrary ordering. Additionally, they cannot represent temporally changing information such as whether or not the shuttle bay doors are open. They implicitly code this by requiring the open door task to precede other tasks. However, if the doors are ever closed for some unscheduled reason, their systems have a difficult time rescheduling. The third weakness of their current approach is that they do not have a interactive graphical interface to their schedule; their process is mainly paper-driven. Finally, they do not have the ability to reactively reschedule; they must start the scheduling from scratch resulting in a completely new schedule that unnecessarily changes much of the schedule. In 1991, we plan to deliver an interactive scheduling tool that is effectively integrated into the Shuttle Processing Data Management System (SPDMS-II). We plan to support this tool for approximately one year, after which we believe it will be officially adopted and supported by the SPDMS-II effort. ARC is developing the scheduling algorithms with assistance from the LAIC. LSOC is supporting the interface with the KSC schedulers and engineers, an activity that entails substantial knowledge engineering.

Milestones:

1990: • Demonstration of KSC Cargo Operations scheduler.
• Multiple demonstrations of the scheduling tool at KSC with increasing functionality (approximately once every two months).
1991: • Evaluation of the scheduling system at KSC.
• Submission of a AAAI paper on iterative improvement scheduling algorithms utilizing a Connection Machine.
1992: • Official deployment of the tool at KSC.
• Development of a distributed scheduler for multiple but interacting facilities.
1993: • Integration of the tool with a planning system.
1994: • Development of a cooperative distributed scheduler for KSC launch processing.

Heuristic Constraint-Based Scheduling

Steve Smith; Carnegie-Mellon University

This work continues the application of scheduling techniques, similar to the ISIS/OPIS systems originally developed at CMU, to the Hubble Space Telescope science scheduling problem. It concentrates on the development of heuristics, specific to planning, scheduling and resource allocation, that improve constraint-based planning and scheduling. A novel component of the work is that it considers a probabilistic analysis of possible schedules.

In the past year, efforts have concentrated upon formulating the HST problem as a constraint satisfaction problem using the ISIS/OPIS framework. This required extensive knowledge engineering and considerable extensions to the ISIS/OPIS framework, including a generalization of time-varying state variables and the incorporation of a planning component.

In 1990, a demonstration of a planning and scheduling prototype was completed. It successfully planned and scheduled a representative subset of the Hubble Space Telescope short term scheduling problem. This work is under evaluation by the Hubble Space Telescope Science Institute and is documented in a CMU Technical Report entitled, "Generating Telescope Schedules."

In 1991, this architecture will be scaled to handle a wide variety of HST scheduling tasks, taking into account more constraints and activities. A focus of this research will be an analysis of the integration of planning and scheduling.

Milestones:

1990: • Completion and demonstration of an initial planning and scheduling prototype.
1991: • Completion and demonstration of an operationally-useful planning and scheduling system for the Hubble Space Telescope.

Constraint Networks
Rina Dechter; UCLA

This work concentrates on the application of constraint network methods to scheduling problems. The principal idea behind the research is that scheduling problems are easily represented as constraint networks. Some constraint networks can be built with a structure that facilitate fast search. However, for typical real world scheduling problems this is not the case. Professor Dechter has theorized that even complex scheduling problems can be broken up into manageable sub-problems which can be isolated, solved quickly, and integrated back into the original problem resulting in a fast overall solution. This claim has been confirmed for test cases; a major goal of this activity is to show that it also holds for complex, NASA scheduling problems. It is likely that this work will use either the STS or HST scheduling domains discussed above.

Milestones:

1991: • Application of constraint networks heuristics to a NASA scheduling task.
1993: • Journal article submitted reporting results.

Neural Net Scheduling (SPIKE)
Mark Johnston; Space Telescope Science Institute

The Hubble Space Telescope science scheduling problem is a combinatoric problem that has proved to be an excellent test of current scheduling techniques. The problem involves scheduling the observations satisfying orbital and resource constraints so that instruments are utilized productively. SPIKE is a constraint-based system that solves this problem using traditional constraint satisfaction techniques and uncertainty mechanisms. This work involves compiling constraint representations into neural nets. The nets embody the global structure of the constrained problem and allow for very fast scheduling. This work involves the continued development of the neural net approach as well as the empirical testing of its efficiency on realistic telescope schedules. SPIKE is already being phased into operational use at STSCI.

Milestones:

1990: • Development of a general neural net synthesizer from a SPIKE representation.
• A general scheduling suite consisting of traditional and neural net

techniques using SPIKE specifications for long-term HST scheduling.

2.2 Human-Machine Interaction

A computational system that serves in the role of a truly intelligent assistant should be able to communicate effectively and at many levels of complexity with its human users. Moreover, such a system should be able to acquire knowledge continuously during its interaction with humans, improving the smoothness of its performance over time. The work described in this section focuses on research into knowledge representation and acquisition methodologies which will facilitate intelligent interaction with humans. We are also exploring the integration of AI-based interface methods with other tools, including hypermedia and virtual reality devices.

Our work is being accomplished in collaboration with personnel of the Ames Aerospace Human Factors Division. They contribute vital expertise in human psychology and modeling as well as experience in the design and construction of a wide variety of human-machine interface devices including the data glove and the helmet-mounted display.

Intelligent Interaction and Knowledge Acquisition

Guy Boy, Jody Gevins; RIA
Steve Ellis, Irv Statler; ARC Aerospace Human Factors Division
Tom Gruber; Stanford University

The goals of this project include the design and implementation of: a knowledge representation methodology, called "Block KR," which facilitates the construction of intelligent interfaces; a knowledge acquisition system that is integrated with the Block KR; and, a user interface generator. In 1990, work has focused on formalizing and efficiently implementing the Block KR as a representation framework for operation manuals, procedures checklists, user guides, and other on-line tools useful for controlling complex dynamic systems. In addition, the method allows for the representation of operational procedures. A knowledge block includes five components: a goal, preconditions, a procedure made up of a set of actions, a set of abnormal conditions, and a context. This basic representation supports the creation and maintenance of abnormal (or recovery) procedures, allowing one to represent the context in which a procedure holds. A formal theory of the Block KR has been completed and the representation has been tested on an intelligent interface into a small but significant subset of Space Station Freedom documentation.

This interface forms the basis of an effective Computer Integrated Documentation (CID) System. When integrated with hypermedia technology, the CID system will observe a user browsing a document, acquire his browsing strategies, and generalize them to produce intelligent indexing mechanisms. A

first demonstration of the system has shown the value of context-dependent index acquisition.

In 1991, work will continue on improving the Block KR and associated knowledge acquisition techniques to provide for effective performance on increasingly larger human-machine interface problems. In the context of the CID system, the techniques will be combined with a hypermedia system. The CID system itself will be evaluated in the context of the Space Station Freedom Technical Management and Information System (TMIS).

Milestones:

1990: • Demonstrate utility of Block KR as part of an intelligent assistant system for analysis of SSF documentation.
• Demonstrate the CID prototype and test it with selected users.
1991: • Integrate CID in real world environment (e.g. TMIS)
1992: • Demonstrate integration of Hypermedia with AI-based enhancements to SSF TMIS.
• Demonstrate CID as an integrated intelligent hypermedia system.
1994: • Utilize machine learning methods to demonstrate adaptive operator faces to existing AI systems.
• Demonstrate CID as an adaptive context-sensitive information media.

Procedure Management and Maintenance
Guy Boy; RIA
Robert Mah, Jay Steele; ARC Advanced Missions Technology Branch
Rick Jacoby; ARC Aerospace Human Factors Division

The goal of this project is to improve the performance of teleoperated remote assembly tasks by developing effective means of working with manipulation procedures. A major problem in using "canned" procedures is the time lag between an operator and an assembly robot. A technique for solving this problem is to first apply a procedure in a simulated or "virtual" environment and only provide actual commands to a robot when the operator is satisfied with his virtual results. Since the real world may not always match the virtual world, a feedback mechanism is needed to warn the operator of abnormal or unexpected results in command execution. Recovery strategies can be tested in the virtual environment and then applied to the actual robot. Automatic knowledge acquisition during this process provides for both improvement of the fidelity of the virtual world and storage of successful fault recovery strategies for later use in analogous situations. In 1990, a Puma robot arm performing simple construction tasks is being used as an experimental testbed; the knowledge representation and acquisition mechanisms described above are used for procedure management, and the virtual reality devices developed by the Ames Aerospace Human Factors Division provide the simulated world environment.

Milestones:

1990: • Demonstrate a mockup of procedures management and maintenance on the telerobotics setup at Ames.
1991: • Demonstrate the influence of human perception on procedures management and maintenance.
• Demonstrate procedure acquisition and refinement according to context.
1992: • Demonstrate the influence of interaction devices on procedures management and maintenance.
1994: • Utilize machine learning method to generalize procedures and design an analogical mechanism for handling procedures.

Developing a Sophisticated Computer Model of Human Behavior

William B. Gevarter; RIA

The approaching era of manned space stations and space exploration carries with it the promise of advanced automation featuring intelligent computer programs and machines. If such systems are to achieve a truly symbiotic relation with humans, they will require sophisticated modeling of their human partners; this is the thrust of our research.

As an initial contribution to this effort, we are developing a computer framework as an aid in understanding and integrating many of the existing theories and findings in motivated cognition. Motivated cognition focuses on the motivations or affects that provide the context and drive in human cognition and decision making. The approach is to first develop, in diagrammatic form, a conceptual architecture of the human decision making approach from the perspective of information processing in the human brain.

In the past year, a preliminary version of a conceptual architecture of human decision making has been developed. This architecture has been utilized as a vehicle for successfully constructing a computer program simulating Dweck and Leggett's (1988) findings (that relate how an individual's implicit theories orient them toward particular goals, with resultant cognitions, affects, and behavior).

Our future work involves seeking out other segments of information on motivated cognition, evaluating this information and using the results to update the framework and computer models discussed in this paper. In addition to further work on affects, it is proposed that belief systems, internalized world models, human decision heuristics, more complex behaviors, and other aspects that reflect human psychological behavior eventually be added to the model.

Milestones:

1990: • Elements of human decision heuristics be incorporated into the model.

1991: • Simulation of belief systems and their effects on attention, selective information retrieval, and belief persistence be incorporated into the model.

3. Scientific and Engineering Data Analysis Tools

3.1 PI-in-a-Box

Silvano Colombano, Michael Compton, Richard Frainier; RIA
Laurence R. Young, Rajiv Bhatnagar, Nicolas Groleau, Sen-Hao Lai, Peter Szolovits; MIT
Chih-Chao Lam; Stanford University
Meera Manahan; Lockheed/JSC

The conduct of experimental science in existing spaceborne laboratories such as Spacelab or future ones such as Space Station Freedom is severely constrained in several respects. The Principal Investigator (PI) is normally not on the spacecraft and communication with the ground is often limited in bandwidth or availability. Because of the open nature of the air-to-ground voice links, free discussion of experimental alternatives is inhibited. Furthermore, the experiment-specific decision making ability of the astronauts is limited by the training they have been able to receive before the flight and by the time they have available in flight. Longer mission durations make it more likely that contingencies will arise for which the astronaut will have had no adequate preparation. Interviews with crew have elicited a strongly-felt desire to have available on-board enough information about complex experiments to enable them to be productive, reactive scientists.

Considering the limited opportunities that exist for flight experiments, and the scarcity of both space and crew time, an intelligent system to assist crew in the conduct of spaceborne science will be useful, especially if it contains much of the experiment-specific knowledge known to the PI. In an attempt to solve these problems, in 1988, Professor Laurence Young of MIT (who had experiments on the Spacelab SL-1 and D-1 missions) began a collaborative project with computer scientists at ARC and Stanford to build a knowledge-based system that can carry on-board much of the knowledge possessed by ground-based principal investigators.

The primary user of the system will be the astronaut performing an experiment, but the PI and the mission manager may also occasionally use the system. The initial emphasis is on real-time consultation between the astronaut and the program, and between the PI and the program, during and shortly after the conduct of an experiment in space. The program serves the following functions, alone or in conjunction with the user: signal quality monitoring, malfunction diagnosis, protocol management, quick look data analysis, recognition of unusual or significant events, suggested protocol changes, and anticipation of additional resource requests.

A specific Spacelab experiment, for which Professor Young is the PI, was chosen as the test case, both because of the experience accumulated and because it is a crew intensive activity. The experiment is the Rotating Dome, a

set of tests of adaptation to weightlessness conducted before and after flight on Spacelabs 1 and D-1, and in flight on the German D-1 mission. It is scheduled for continued testing on two more Spacelab missions and for development as a possible Space Station Freedom experiment.

During 1990, we concentrated on three areas: the Protocol Manager, the Interesting Data Filter, and the overall system architecture, with emphasis on the interfaces between system modules.

A Protocol Manager prototype has been completed of sufficient complexity to demonstrate the possibility for a decision making process that would have been unlikely without the presence of a PI. A new protocol is produced and proposed at the request of the astronaut/experimenter whenever a change is necessitated by lack of time or faulty instruments, or when interesting data is detected.

The Interesting Data Filter helps to understand whether numeric data matches the output that is expected from the experiment instruments. The strategies considered at this point are based on discrepancies from expected results, where the expectation takes into account a model of adaptation, and is based on results previously obtained from the same experimental subject or, when the subject is new, from other subjects.

The remainder of 1990 will be spent in preparing for ground test during the SLS-1 mission scheduled for August, 1990. The system will assist the ground-based experimental staff in real-time data analysis and equipment diagnosis. In addition, it is likely that we will flight test a Macintosh Portable on the SLS-1 flight; we believe this particular hardware is a very strong candidate for spaceborne use on the SLS-2 mission.

In 1991, based on the results of the SLS-1 ground test, we will continue developmental work on all of the PI-in-a-Box modules. Of particular importance will be work on flight hardware and software in preparation for the 1992 SLS-2 mission. In addition, we will begin integrating work in automated discovery conducted in RIA into the Interesting Data Filter, thereby expanding its current rudimentary quantitative analysis capabilities.

Milestones:

1990: • Conceptual Design Review
- Develop software for ground test.
- Test of Pre-flight ground system.
- Ground test during SLS-1 flight.

1991: • Develop flight hardware and software.

1992: • Flight of PI-in-a-Box system on SLS-2.

3.2 Automatic Classification and Theory Formation

The main goal behind automatic classification and theory formation is the modeling of observed phenomena. We would like to build computational systems which are able to discover new information about phenomena from the systems' prior knowledge and observations.

Our activities in this area include both analytical and empirical techniques. With the analytical approach, the system begins with a strong theory of the expected phenomena and then extends and modifies this theory according to its observations. These observations can be of external phenomena or generated through an experimentation process. Alternatively, the empirical approach takes many example observations and then forms a probabilistic model of the data. Generally speaking, these empirical techniques have very weak prior knowledge.

This line of research is extremely important to NASA for two main reasons: 1) there is a voluminous amount of partially interpreted scientific data gathered from NASA missions; 2) long-term missions will require systems to discover new concepts and to integrate these concepts with an existing theory as unexpected phenomena are witnessed.

Bayesian Learning (AutoClass)

Peter Cheeseman, Wray Buntine, John Stutz, Robin Hanson, Bob Kanefsky, Farhad Towfiq; RIA

Bayesian learning is a form of statistical learning that is particularly good at finding patterns in noisy data. In addition, Bayesian theory can guarantee that the patterns found represent a real effect operating in the data, and not an artifact of the data analysis. It is also well suited for using prior knowledge and statistics to guide the selection of patterns among competing alternatives. While the general Bayesian theory has been around for many years, the discovery of its implications in particular domains is still largely unexplored.

As an important domain that illustrates the Bayesian approach, we developed the theory as applied to automatic classification of data. As a result of initial exploratory work, we implemented an automatic classification program (AutoClass II) that is sufficiently mature that it can be applied to many different databases. AutoClass II can find classes with a combination of real-valued and discrete data and no prior information about what classes might be present. Unlike previous automatic classification programs, AutoClass II does not need to be told how many classes are present or even if there are any classes at all. The system uses a new extended Bayesian approach that searches for the most probable classification, and assures that any classes that are found have a real cause--i.e., classes are not an artifact of the search process.

NASA has many largely unanalyzed databases which could benefit from this type of automatic classification. To test the utility of our work in Bayesian classification, we applied Autoclass II to several real databases. The largest

database analyzed was the IRAS Low Resolution Spectral database, containing infrared spectra of over 5,500 stars. The resulting classification included many well-known classes of stars as well as new classes of subtly different spectra of considerable astronomical interest. Independent information has since confirmed the validity of these new classes. This new classification was published jointly with infrared astronomers as NASA Reference Publication #1217 in March 1989.

As a result of these data analysis experiments, a number of limitations of the original AutoClass II system became apparent. In particular, the current system assumes independence of the attributes within a class for simplicity, but this assumption is a major limitation in its performance. These experiments showed that a major improvement to AutoClass II would be allowing various models of dependency to be incorporated into the search. Beginning in late 1989, we began work on methods for discovering significant correlations between variables, and building these correlations into the classification model.

The theoretical research on discovery of significant correlations in data turned out to be much more complicated than was originally expected. However, in 1990 we have made considerable progress in modeling the major forms of correlational dependency. This research is a first step on extending AutoClass systems to handle attribute dependency.

In parallel with the development of dependency models, we have been rewriting the original AutoClass II code to make it faster, more accurate, and more user-friendly. The process of turning experimental code into a robust usable system is a larger effort than the original code development. However, a new version of the system, AutoClass III, is now being integrated with the input interface, and will be available for general release in 1990. Additional research in 1990 and 1991 is aimed at finding hierarchical classes, and integrating this capability with the discovery of inter-variable dependencies within an efficient search procedure.

In addition to extensions to AutoClass, Bayesian theory can be extended in various ways that provide useful data analysis tools for NASA. Dr. Wray Buntine has recently joined the group, to do research on the discovery of patterns that predict particular variables (supervised learning). This research is important because in many NASA applications, the goal is to make specific predictions rather than look for relationships among a set of variables, as in classification.

Milestones:

1990: • Extend AutoClass to include correlations between variables within a class, and allow hierarchical classifications.
1991: • Apply Bayesian learning to time dependent data.
1992: • Short range predictions for simulated chaotic systems (i.e. finding the equations governing the system behavior and using these equations to make predictions).

1993: • Application of chaotic learning (developed in 1992) to real data 1994:
• Extension of previous work on time series prediction to include large scale multivariate analysis.

Efficient Learning Algorithms
Phil Laird, Evan Gamble; RIA

For NASA, an important objective of machine learning research is to make it easier to produce the vast volume of software needed to monitor and control hardware that will be in space for decades, often out of range of direct human control. Since most of this software is extremely detailed and complex, the strategy is to construct the system so that it adapts to changing conditions by learning, instead of programming it in advance to handle every conceivable contingency. Current techniques for doing this are basic but promising; they need to be evaluated and extended, and new techniques need to be developed.

For machine learning to progress from a collection of ad-hoc programming tricks to a reliable and useful engineering discipline, the algorithms for learning have to be carefully designed and well understood. The mission of the Efficient Learning Project is to focus on the formal specification and mathematical analysis of learning algorithms that are both practical and efficient. This includes the analysis of existing algorithms that others have proposed but whose properties are poorly understood, as well as the development of new algorithms specially targeted to solve problems required by NASA applications.

In the two years of its existence, the project has obtained both kinds of results. The CDFI algorithm is a new algorithm for unsupervised learning. It takes sensor readings from an arbitrary device, and by analyzing those readings, it makes a "map" of the normal operation of the device. This is useful when the device begins to malfunction, because when the readings are found to diverge from normal, the control program can alert a human operator (or expert system) to the divergence and the rate at which the divergence is occurring. The power of this algorithm is its generality; no specific knowledge about the device is required. The CDFI algorithm can be used as part of a monitoring package for many different systems. This work was completed in 1990 and is now available technology. We are in the process of exploring possible applications, such as in the control of closed ecological growth systems in space.

Other research in 1990 and continuing in 1991 focuses on the problem of algorithmic methods, whereby computer programs get faster with practice as they solve problems. This is in contrast with the current situation where most programs, given the identical problem or a series of closely related problems, will follow the same procedure toward a solution every time, never remembering or benefitting from prior experience. New techniques for this type of learning, as well as theoretical analysis that simplifies our understanding of existing techniques, have been developed during this past year. The theory is

in the process of being published, and experiments to test its usefulness are being designed.

Milestones:

1991: • Continue developing analytic learning algorithms and testing them on artificial problems. Continue exploring NASA machine learning requirements and evaluating available tools for satisfying them.
1992: • Apply CDFI Learning algorithm to help predict alarm conditions for scientific experimentation.
1993: • Demonstrate applicability of unsupervised learning to a closed loop process control system.

IL Theory Formation Project
Michael Sims; RIA

Theory formation, the initial formulation and subsequent modification of theories about the world, is the crux of intelligent behavior by an agent. In the IL theory formation project we have concentrated on theory formation in domains which have rich, formal (mathematical) theories such as the physical sciences and engineering domains. An example is how we formulate and modify a theory of some physical object where we already understand much of the fundamental physics or chemistry involved. This can be contrasted with Cheeseman's and Laird's work where one begins with a minimal theory about the domain objects.

There have been two foci of this project: (i) the integration and control of the theory formation process, and (ii) theory driven concept formation. The former focus has led to the development of an exploration discovery architecture, called IL, which allows for a declarative representation of the domain objects, heuristics and relations and for the ability to deduce consequences from the domain's formal theory. The IL architecture has permitted the integration of elements for handling experiments (examples), deductions, conjectures, supporting conjectures, and formally validating results into coherent and realistic reasoning scenarios. This approach has been tested on a number of specific case studies.

The second focus, theory driven concept formation, has led to a study of purposive operator discovery. Humans typically have a purpose in mind when they attempt to create or discover a new mathematical operator, and our work in purposive operator discovery utilizes an object's intended purpose to aid in its discovery. We use a specialization of generate and test which we call Generate, Prune, and Prove (GPP). GPP generates possible candidates, then prunes those candidates by applying them to examples. Only if an operator passes this empirical prune phase does GPP attempt a formal proof that the operator is appropriate. During the prune phase there is typically only partial information about many of the objects involved, so for GPP to be successful it is necessary to do partial evaluation of the purpose constraint. We have

successfully applied the GPP methodology to the automatic discovery of the Conway multiplication operator, a known, yet difficult discovery.

In 1989 and 1990 we have concentrated on the modeling of pure mathematical systems because that allowed the simplification of many of the difficult problems involved with realistically rendering a real world domain. In 1991 and beyond we will focus on the application to real physical objects and systems. In particular, the work is being refocused on models of planetary atmospheres.

It is useful to emphasize two limitations of previous work in the field which will be addressed in this refocused work:

(a) need for a higher level language for representing a theory
(b) need to utilize contextual knowledge

In the domain of atmospheric physics, the fundamental objects that physicists deal with are equations of physics, abstract data sets, and control schemas. It is very difficult to deal with these objects at an appropriate level of abstraction in a programming language such as FORTRAN. As a consequence, theories with straightforward descriptions to a physicist are difficult to modify and manage on a FORTRAN program level. However, these FORTRAN programs are an instantiation of such a physical theory, and if we had a representation language at an appropriate level of abstraction, it would facilitate the building and manipulation of such theories. Therefore, our first priority is the development of an appropriate high level structure for handling such theories on a computer.

One insight gained from applying IL to mathematics is that there is something fundamentally important about the difference in representation of objects by humans and what we usually do on computers. Humans keep much of their knowledge in a contextual or implicit level and only explicitly represent that knowledge as needed. For example, when a physicist says "F = ma," he seldom states that the mass, m, is the same object which has acceleration, a ("same object referenced" assumption). On the other hand, when we program these in a machine we initially allow for all of these possibilities and encode such connections from the beginning. The problem is that when we make so much information explicit it clutters other processing, and in general makes processing more difficult and adds to the computational complexity. As a first step, we will address the problem of robust transitions between implicit and explicit knowledge for data structures.

Milestones:

1990: • Develop high level representation language for atmospheric physics models.
1991: • Develop mechanisms for use of contextual knowledge for atmospheric physics data structures.
1992: • Integrate representations of atmospheric models and reasoning context with existing IL inference mechanisms.
1993: • Demonstrate IL exploring the space science domain of planetary atmospheres with all of its discovery mechanisms operational

and integrated.

Atmospheric Modeling
Rich Keller, Michael Sims, Esther Podolak; RIA
Chris McKay; ARC Theoretical Studies Branch

Planetary physicists commonly use complex numerical models (usually implemented in the form of large FORTRAN programs) to help predict and analyze the composition of planetary atmospheres. Unfortunately, a thorough understanding of the technical details of the program implementing the model are required before it can be used effectively to solve a problem. Often, this means that the original programmer must be intimately involved in running the model to solve a new problem. In addition to simply running the model, planetary scientists frequently have the need to create variations of this basic model. These variants incorporate different underlying assumptions about an atmosphere, leading to different calculations in certain portions the model. For Titan, the current model incorporates a relatively simple two-stream model of radiation transfer within its atmosphere. However, to study more subtle radiation transfer problems, scientists need to incorporate a more sophisticated n-stream model. Currently, the original programmer is the only one who can make this type of modification with any degree of facility.

Our intention in 1990 is to make the Titan model available for use by a larger group of planetary scientists. Specifically, we are building an intelligent graphical interface that permits scientists to inspect an atmospheric model, modify parts of the model, execute the model, and perform analyses on the results. To build such an interface, we are making use of object-oriented modeling techniques to develop a high-level atmospheric modeling language that sits at a level of abstraction above the FORTRAN code level. The objects in the language correspond to domain concepts, physics equations, and datasets that are familiar to the scientist, such as "saturation point," "ideal gas law," and "Voyager refractivity dataset".

As a starting point, we have chosen to focus on a portion of the Titan model that determines gaseous composition, and to develop a modeling tool covering this part of the model. Based on our experience, in 1991 we will extend and generalize our approach to cover other parts of the Titan model, as well as other types of models in other domains. Although the modeling tool will initially take the form of an intelligent machine assistant to the human modeler, we expect to gradually incorporate more autonomy into the tool to ease the burden of the modeler, wherever possible.

Milestones:

1990: • Model analysis: Analyze current Fortran model and determine what types of model changes should be supported by the modeling tool.
• Knowledge representation: Develop modeling primitives and

representation for domain concepts, physical quantities, physical equations, physical constraints, and modeling assumptions;
- Interface: Design graphical interface to tool that facilitates easy construction and modification of Titan gas composition model.

1991:
- Validation: Planetary scientists use model construction tool to conduct numeric experiments on atmospheric composition and stability.

Concept Formation In Classification and Planning
Douglas Fisher; Vanderbilt University

This research project explores extensions to Fisher's earlier work on Cobweb, a system that organizes probabilistic concepts in a hierarchy, retrieves appropriate concepts by sorting observations through that hierarchy, and acquires knowledge by modifying the structure of the hierarchy. At each level of the hierarchy, Cobweb uses an evaluation function based on information theory to select the node that best matches an instance, incorporates the instance into that node's probabilistic description, and recurs to the next level. The approach is inherently incremental in that it interleaves learning with the classification process. At any stage, Cobweb can make predictions about missing features of the objects it observes, and experiments have shown that -- for many classification tasks -- the system rapidly converges on predictive concept descriptions.

In 1990, Fisher has augmented Cobweb's control strategy in several ways. First, he has adapted noise-tolerant learning techniques to identify the optimal level in the concept hierarchy for prediction accuracy. Experiments with two variations of this scheme have shown that one can avoid "overfitting" and significantly improve prediction accuracy over a method that simply classifies objects to terminal nodes in the hierarchy. A related line of work has focused on extending Cobweb to form directed acyclic graphs instead of trees. This lets the system capture orthogonal classes in a concept hierarchy (e.g., mammals, birds, and reptiles vs. carnivores, omnivores, and herbivores). Such redundancy is important when the training data contain many missing attributes and when known attributes support classification only along certain dimensions. Potential applications include diagnosis of physical systems, such as those used in controlling the space station and planetary habitats.

In work begun in 1990 and continuing in 1991, Fisher is adapting ideas from Cobweb to the management of search control knowledge. The basic approach involves carrying out concept formation over explanations and problem-solving traces. Problem solving becomes a matter of classifying a new problem and predicting parts of the solution trace as one descends the concept hierarchy. Noise-tolerant methods identify nodes at which one should abandon classification problem solving and complete the solution using domain knowledge. Preliminary experiments reveal an important and intuitive point that has not been identified in the literature on explanation-based learning: that use of learned rules should depend on the current status of problem solving (e.g., diagnosis of an aircraft component should be guided by earlier diagnosis of

antecedent components). Current approaches to the utility problem apply learned rules based on heuristics that are not informed by the problem-solving context.

In new work starting in 1991, Fisher will apply a similar strategy to the organization of plan knowledge. However, the emphasis here is on overcoming inconsistencies between the projected effects of operators and the environment they are intended to model. In the short term, inconsistencies require efficient methods for replanning that overcome unanticipated operator effects. In the long term, as regularities in actual behavior become apparent, the system supplants its initial operator models with ones that better describe the behavior of the world. Such capabilities are essential for intelligent agents that operate in novel domains, as would occur in planetary exploration.

Milestones:

1990: • Finalize a version of Cobweb that organizes its conceptual knowledge using directed acyclic graphs and carry out systematic experiments to evaluate its behavior in both natural and artificial domains.
• Extend the current work on concept formation over explanations, which relies on logical representations, to employ the probabilistic representations pioneered in Cobweb.

1991: • Initiate systematic experiments with the planning system, investigating the effect of noise (unanticipated operator effects) on planning efficiency and the ability to change operator models in response to these outcomes.

1992: • Extend Cobweb-based planning system to control a complete agent, and begin evaluating the system in an autonomous agent testbed.

1993: • Complete evaluation of Cobweb-based planning in an autonomous agent testbed and begin adaptation to planetary rover or teleoperated device.

1994: • Incorporate Cobweb-based planning methods into a prototype rover or teleoperated device; begin initial field tests.

4. Onboard Systems for Diagnosis, Planning, and Intelligent Control

4.1 DTA/GC (Differential Thermal Analyzer/Gas Chromatograph) Analysis and Control System

David E. Thompson, Rich Levinson; RIA
Rocco Mancinelli; ARC Solar System Exploration Branch

This project is developing AI-based software that autonomously controls a new geochemistry laboratory instrument. This instrument combines two well-known materials analysis methods: differential thermal analysis (DTA) and gas chromatography (GC). Currently manufactured laboratory DTAs require a great deal of human supervision and decision-making during use. The proposed software will enable autonomous operation of a DTA-GC, relieving the laboratory staff of the need to constantly attend the instrument and permitting the use of this instrument in remote or hostile environments. Although the coupling of these analysis systems will result in a more detailed characterization of mineral and soil samples, their combined use will require the development of expertise detailing how the two areas interrelate. Therefore, the software being developed will also provide a framework to acquire and store data relating the testing of hypotheses and the results of experiments. This framework will greatly expedite the acquisition of a comprehensive mineralogic database which will show how thermal characteristics and phase change events relate to chemical composition. Although the project is required to primarily support autonomous control of the coupled system, data analysis modules are also being developed which can be used in the near real-time recognition of interesting events that cause the re-programming of the heating function during an experiment. This recognition requires the comparison of actual events with expectations which are generated before or during an experimental run. The selection of an efficient recognition methodology and an appropriate characterization of an event will utilize machine learning approaches.

A differential thermal analyzer is a programmable oven -- it heats up soil samples at a controlled rate. The heating causes the soil to undergo thermal events such as phase changes in the mineral structure (or even melting or gasification) and release of gases that are trapped in the lattice structure of the particular minerals. The character of such an event (its duration, strength, and onset temperature) is indicative of the mineral structure, the proportion, and the content in the soil. A gas chromatograph is a column of material through which gas mixtures are passed for constituent identification. When a mixture flows through the column, the individual gas constituents separate and can be identified chemically according to their relative flow rates. The coupled system physically attaches the GC to the DTA so that when gas is released from the soil sample, it is evacuated to the GC column and analyzed. This allows the scientist to determine both structural and evolved gas chemistry of a single sample. Both sources of information can be combined for a more complete and

unambiguous characterization. Thus, reliable decisions can be made as to the amount and type of minerals that are present in the soil sample; if the sample is an unknown, then its combined "signature" is compared to characterizations in the database and with expectations generated by the system, and a hypothesis of the components is generated as to what the sample is made of. Hence, the software being created will recognize the characteristic thermal and evolved gas event signatures and will postulate identification of mineral families in the soil samples. It will then suggest and control variations in the experiment run that will help verify the soil composition by eliminating alternative hypotheses.

As a spin-off benefit, the autonomous operations technology and the domain learning elements in the project will allow the DTA-GC to be advocated as a soil sample analysis instrument of choice for future planetary exploration missions. The system can perform analysis either for toxics or for target minerals.

This project began in early 1990. To date, significant work has been accomplished in both geochemistry and software development aspects of the project. The geochemists have begun operation of the DTA on standard reference samples to learn the peculiarities of the instrument and to standardize sample preparation techniques. They have also concluded a period of testing and calibration of the system, begun plumbing the GC connections, and have carried out extensive literature review as to what signatures in mineral families they hope to be able to reproduce. On the software development side, a full conceptual model has been created using the XTK language, a tool developed by the SADP projects as an add-on to the KEE knowledge base development shell. This model consists of task trees by which actions in the instruments can be controlled and data can be extracted for analysis and for generation of temperature ramping functions. Meanwhile, the team has begun to gain some familiarity with the DTA by participating in the experiment runs. This will facilitate knowledge acquisition and representation as the experimental data is loaded into the knowledge base. Additionally, the team has begun working on a representation language for describing the main elements of a thermal event signature so as to have some idea of what must be analyzed in order to ascertain that an event is the same event as one expected, and so that one can select recognition algorithms which will allow both event definition and discovery of unexpected relations.

The actual experiment plan is now in place and consists of a series of runs over five to six months compiling data on clays, sand, clay/sand aggregates, carbonates, salts, carbonates/salts aggregates, and combinations of all of these. At the end of this period, the software team will stop porting data into the simulation system and begin refinement and variations of the model. Essentially, the team will consider the database complete, even though the geochemists will continue to look at more complex assemblages such as organics mixed with the salts and carbonates.

Once the simulated system is capable of controlling a complete experimental run and achieving "correct" answers as to the mineral assemblages in the sample, it will replace the computer system which interfaces with the user on the

DTA, and new experiments will be run to test the software capabilities as an autonomous controller. From that point forward, work will concentrate on fault diagnosis and control of the physical components, and on analysis of unexpected events through refinement and variation of the analysis modules.

Milestones:

1990: • Completion of conceptual design.
• Implementation of software control of DTA-GC simulator and application to interpretation of simulated experimental results.
• Completion of knowledge base.
1991: • Enabling of initial control of physical device.
• Final demonstration and delivery.

4.2 Superfluid Helium On-Orbit Transfer (SHOOT)

Timothy Castellano, Jeff Shapiro, Frank Robinson, Gary Villere, Eric Raymond; RIA

The SHOOT experiment is a shuttle payload bay cryogenics experiment scheduled to fly on STS-57 in August of 1992 on the orbiter Endeavour. The experiment is a joint effort between the Goddard Space Flight Center Cryogenic Technology Branch and the Ames Research Center Artificial Intelligence Research Branch. The SHOOT principal investigator is Dr. Michael DiPirro of Goddard.

Ames' role in the SHOOT project is to provide computer hardware and software to control and monitor the experiment through all operational phases. These phases include ground testing, servicing on the launch pad, in-flight control from the Payload Operation and Control Center and from on board the shuttle Aft Flight Deck.

The SHOOT experiment apparatus consists of two insulated containers of Helium (dewars), readout and control electronics and a support structure. The two dewars are connected by a vacuum insulated transfer line that is removable, much like a service station hose. The helium is normally in a state called "Superfluid" in which it flows without viscosity. The main experiment objective is to demonstrate the ability to transfer this unusual liquid from dewar to dewar, much as would be done during a cryogenic servicing of a cooled orbiting telescope such as the planned Space Infrared Telescope Facility (SIRTF).

Infrared telescopes such as SIRTF need cooling of their optical components in order to reduce background radiation to enable detection of faint sources of heat radiation. The phenomenally successful IRAS (Infrared Astronomical Satellite) had an on-orbit lifetime of about a year limited by the total amount of helium that it was launched with. If future satellites are designed with resupply capability they may have lifetimes of up to 10-15 years. The science return can then be that much greater, maximizing the return on investment. To achieve this,

a helium resupply capability of up to 10,000 liters has been imagined and preliminary studies of this so called Superfluid Helium Tanker have been funded by Johnson Space Center (JSC).

SHOOT will determine if this is technically possible. Software being developed will provide facilities for the control and monitoring of the experiment as each hardware element and procedure are exercised. In addition, to support the transfer of 10000 liters of helium, over many hours during a nominal tanker operation, requires some on-orbit knowledge of the transfer process. This knowledge could be provided by either a highly trained crewperson or be contained in the facts and rules of an onboard expert system. During any future resupply mission crew time will be, as always, at a premium. Unfortunately, the shuttle's 5 IBM computers are flight critical and not designed to support an expert system approach. Fortunately, JSC realizes the limitations of the Shuttle General Purpose Computers (GPC) and is exploring a policy where payloads may use a flight qualified GRiD 1530 80386 microprocessor laptop machine for performing monitoring and control tasks.

SHOOT will take advantage of this technology by developing programs tailored to the specific control and monitoring functions necessary to achieve experiment objectives. One of these objectives is to demonstrate a Zero-G fluid coupling operation. An astronaut will couple and uncouple the helium transfer line during a planned Extra-Vehicular Activity (EVA). A second crewmen in the orbiter cabin will monitor the condition of the SHOOT dewars on the screen of the GRID using a program developed by RIA. The GRID will be connected to the SHOOT experiment electronics and will receive experiment data and have the ability to control the experiment for several predetermined protocols selected by the Principal Investigator.

A helium transfer will also be conducted without ground intervention. The AFDEX (Aft Deck Expert) will assume control during a predetermined phase of the experiment and will provide a shuttle crewmen with advice and procedures for an efficient helium transfer operation. Several possible failure modes will also be diagnosed if they occur. Corrective action will be taken by the GRiD upon astronaut concurrence.

The expert system was written using CLIPS (an expert system shell developed at NASA JSC and available through COSMIC) and the C language under MS-DOS. It contains about one hundred rules and 114 initial facts. It runs interactively and provides graphic and text displays for the astronaut's use.

For the AFDEX to be successful it must contain all the expert knowledge of the SHOOT Principal Investigator and display it in a format that is familiar to the astronaut crew. This addition to the capability of the Space Shuttle uses computers to put humans back "in the loop" of onboard scientific investigation.

Other responsibilities of the SHOOT software staff include supporting payload qualification testing, development of simulation software to train the astronaut crew, operation of the payload at Kennedy during post shipment checkout and

integration, and from the Goddard Payload Operation and Control Center during flight. In addition, the SHOOT payload requires a final top off of helium about two days before launch. This operation requires miscellaneous pumps and helium tanks and the ground support equipment computer hardware and software to be brought to the launch pad. The SHOOT software staff will assist in this operation.

Milestones:

1990: • Second laboratory dewar test. Flight-like operations performed. AFDEX performs nominal helium transfer.
• Testing begins with flight hardware.
1991: • Astronaut crew training begins with AFDEX and payload simulator.
• Final ground and flight software configuration and delivery.
1992: • SHOOT is launched on STS-57 on the new orbiter Endeavour

4.3 Exploration Technology Planetary Rover - Operations Autonomy

David Thompson, Michael Compton; RIA
Keith Swanson; ARC Advanced Missions Technology Branch
Stan Rosenschein, Nathan Wilson, Leslie Kaelbling; Teleos Research

The objective of the Exploration Technology Planetary Rover - Operations Autonomy project is to develop software and methodologies by which an autonomous or semi-autonomous rover can instantiate activities on a planetary surface through the development and selection of plans and allocation of resources. Specifically, 1) to develop the software which supports and controls robotic activity such as navigation, mobility, and communication in support of high level goals for the rover, and 2) develop software which schedules and plans sample acquisition strategies on the Moon, Mars, or other planetary bodies.

This project is being carried out as a joint development effort between NASA Ames and Teleos Research, a private research and development company with expertise in artificial intelligence, reasoning, machine vision, natural language processing, and robotics.

Interactions between various rover subsystems (e.g., navigation, science, communication, power, and mobility) are complex and cannot be fully predetermined. Technology development in the area of interacting planning systems offers the potential for maximizing mission return for a semi-autonomous robotic explorer, as well as for piloted or teleoperated rovers where some of the interactions will be with human operators.

In 1989, an architecture was developed which utilizes an on-board constraint propagation mechanism to generate and maintain timelines integrating multiple interacting plans. This architecture allows subsystems to present plans to a central mediator for incorporation into a timeline. As the timeline is executed, new plans can be presented for scheduling, and goals and plans can be

interrupted, abandoned, or preempted. This architecture also allows for the generation of plans that require input from multiple subsystems, enabling subsystems to cooperate efficiently to achieve complex goals.

In addition to developing an architecture and identifying a new technology to be modified for incorporation into the interacting planner task, a Design Reference Mission (DRM) was prepared. This DRM, which describes a typical day in the life of a planetary rover in great detail, has been made available to the project team and will be used a source of detailed mission scenarios for testing this technology in the following three increasingly-realistic environments:
1) a 2-D simulator of a rover traverse with interacting goals; 2) an indoor robotic vehicle with simplified perception tasks, and 3) the JPL integrated testbed.

Current plans call for the mutual interaction with the JPL Operations Autonomy team, whereby Ames will use the JPL simulator as the initial test facility, or JPL will work collaboratively with the Ames and Teleos personnel so that a single, functional simulator can be developed for the program. Ames/Teleos intends the system to exercise navigation and science planning goals which represent the most interactive and competing goals. The subsystems will supply plans to meet those goals. As the rover moves through the simulated environment, goals can be modified by the user, and new goals will be added by the system itself. For example, as the rover is moving to a new location, if it comes into a scientifically interesting environment, a science alarm will be triggered and a plan developed and presented to the mediator. This plan and other current plans will be mediated by the constraint propagator, resulting in a selection of options.

The next step will be to move the system onto an existing vehicle with simplified perception requirements. Finally, the system will be integrated into the JPL outdoor testbed. This will allow the system to be tested on a rover prototype in an outdoor environment in conjunction with the actual navigation system being developed at JPL.

Milestones:

1990: • Develop detailed description of proposed rover executive.
- Develop simple 2-D simulation scenarios involving multiple competing/cooperating subsystems.
- Extend simulation to include stubs/mock-ups of at least two different subsystems (e.g., science and communications).
- Implement breadboard version of executive architecture.

1991: • Develop detailed subsystem interface specifications .
- Enhance common representation of tasks, goals, and constraints.
- Test simulator with increased levels of fidelity.

1992: • Demonstrate rover executive on indoor mobile robot.

1993: • Demonstrate rover executive on JPL integrated testbed.

4.4 Intelligent Interacting Agents

This research is concerned with the generation and execution of agent activities (we consider an agent to be any active process involved in the execution of some task -- a computational reasoning system, a piece of machinery, a robot, or a human). Lansky's GEMPLAN project is concerned with the problem of generating and executing coordinated multiagent activity in an efficient fashion that also pays close attention to problems of correctness (i.e., adherence to domain constraints). The Stanford/Teleos efforts are more concerned with the problems of a single agent in a highly changing environment. Here, correctness is less of an issue than appropriate reactivity, and the work is highly applicable to robotic domains. Drummond's ERE project fits between these two efforts -- attempting to generate activity (primarily for single agents) that is both reactive and goal-directed. It attempts to focus computation on the most probable courses of affairs, thus mitigating computational complexity with a probabilistic approach. The efforts at CMU and Kedar's project both deal with utilizing learning to improve the planning process and reactiveness. The CMU effort is concerned primarily with improving the efficiency and coverage of planning techniques as well as improving unplanned reactions. Kedar's work is also concerned with adaptively improving control of reactive functions.

GEMPLAN Multiagent Planner

Amy Lansky; RIA

This work focuses on the problem of generating multiagent plans for domains with complex coordination requirements. Thus, it deals with both action generation and action ordering, as well as scheduling issues such as resource allocation and timing. Over the past year, the primary activity on this project has been further development and enhancement of the GEMPLAN multiagent planner, which Dr. Lansky originally developed at SRI, International. The system is now a fully domain-independent multiagent planner, with capabilities exceeding those of many state-of-the-art planning systems.

Work on GEMPLAN in 1990 has two major thrusts. The first is exploring the use of "locality" or domain structure to partition domain information as well as the planning space. Representationally, locality can be used to help handle the frame problem, by explicitly limiting the applicability and scope of effect of constraints and events. More importantly, localized reasoning provides a way of alleviating the costly nature of planning (especially multiagent planning) by partitioning the planning search space into smaller, localized search spaces, and thus may facilitate planning in large domains. GEMPLAN's localized search algorithm handles not only hierarchically partitioned domains, but domains with regional overlap as well--a case that is more complex due to the need to maintain consistency between search spaces.

The second focus of this project is devising new methods of constraint satisfaction, to enhance GEMPLAN's repertoire of planning capabilities. GEMPLAN users write their domain description in a general-purpose

specification language that enables the use of a variety of different kinds of constraints. These domain specifications are then "compiled" into constraint satisfaction code. Several new constraint forms have been added (and previously implemented constraint forms have been extended) to form the following set of GEMPLAN constraints: constraints on patterns of behavior expressed as regular expressions, a variety of temporal and causal constraints among events, a full implementation of the modal truth criterion for multiagent plans (the attainment and maintenance of state-based conditions), and the decomposition of nonatomic events into patterns of subevents. Nonatomic decomposition, in particular, is done in a way that is more general than in standard hierarchical planners -- the nonatomic events are retained within a plan even after they are expanded, enabling reasoning to occur at multiple levels of abstraction within the same context, as is appropriate for each particular constraint. A metric temporal reasoning facility (similar to Dean's time map manager) has also been implemented, but has not yet been integrated into the current GEMPLAN framework.

A recent topic of interest is the notion of run-time constraint satisfaction; that is, satisfying certain kinds of constraints during plan execution, and in a way that maintains plan correctness. This will be especially useful for achieving priority requirements on resources. Such requirements are important in multiagent domains, which tend to resolve resource contention using a run-time priority policy. This subject is also related to the notion of disjunctive plans (plans that have multiple possible execution paths) and the resolution of some forms of disjunction at run-time.

GEMPLAN has been applied to several example problems. Of course, it solves multiagent blocks world problems. The Tower of Hanoi problem is optimally solved with no clues about solving subproblems. (We have both single-agent and multiple agent solutions for this problem, but the single-agent case is definitely the most difficult case for this particular problem). We have also applied GEMPLAN to a small construction domain example (forty-nine actions are generated). This domain includes multiple walls and contractors, and thus requires both resource allocation and coordination of actions occurring within shared regions. This was a useful test of our new localized search code, which can handle regional overlap. We have performed various empirical tests experimenting with different levels of localization and regional overlap in this domain, and results have shown that locality provides great efficiency benefits.

For 1991 and beyond we plan to focus on two primary goals. The first is the application of the current GEMPLAN system (in Quintus Prolog) to a NASA domain. This will obviously also involve extensions to the current system. The second goal is the development of a new GEMPLAN system (probably in Lisp) that investigates several new areas: the integration of preplanning and prescheduling with dynamic run-time planning and scheduling mechanisms. (there is potential, for example, for integration with Zweben's scheduler); parallel search of independent localities; and, associative attachment and tracking of constraints with planned events (the current GEMPLAN system does include a limited capability of this kind).

Milestones:

1990: • Complete initial version of GEMPLAN.
1991: • Apply current GEMPLAN to a NASA domain.
• Complete initial design of GEMPLAN-2.
1992: • Complete initial implementation of GEMPLAN-2.
• Complete initial design of integrated planning/scheduling/execution architecture.
1993: • Design and experiment with parallel search in GEMPLAN-2.
• Complete initial implementation of integrated planning/scheduling/execution architecture.
1994: • Design and experiment with learning integrated into GEMPLAN-2.

Planning, Scheduling, and Control: The Entropy Reduction Engine
Mark Drummond, John Bresina, Rich Levinson, Andy Philips; RIA
Nancy Sliwa; ARC Intelligent Systems Technology Branch
Keith Swanson; ARC Advanced Missions Technology Branch

The Entropy Reduction Engine (ERE) project is a focus for research on selecting and scheduling actions in a way that takes seriously the likelihood of action execution failure. There are two main subgoals for this research. First, we are doing theoretical and applied work to integrate the representations and methods of AI planning with those of classical scheduling. Our second research subgoal is to make sense of planning and scheduling in terms of modern discrete event control theory.

Traditional AI planning deals with the selection of actions which are relevant to achieving given goals. Various disciplines, principally Operations Research, and more recently AI, have been concerned with the scheduling of actions; that is, with sequencing actions in terms of metric temporal and resource constraints. Most of this work in scheduling remains theoretically and pragmatically disconnected from planning. By integrating action selection and action sequencing we expect to be able to provide a coherent theory of planning and scheduling that can be directly implemented as useful software tools.

Most planning and scheduling work assumes that the job of the system is done when a plan or schedule has been generated. This view is hopelessly optimistic since actions, once selected and sequenced, often fail during subsequent execution. In the ERE project, we view the business of planning and scheduling as that of controlling the behavior of an environment to satisfy certain user-specified goals. A planning or scheduling system cannot simply produce a plan or schedule and then vanish; the system must instead persist in its attempt to drive the environment's behavior in goal-achieving directions. Under our view neither planning nor scheduling can be a single-shot effort, but rather a process of participation, where the system must attempt to guide and coerce the environment to conform with given behavioral constraints.

We divide the overall system into three components: reaction, projection, and problem reduction. The reaction component is responsible for producing behavior in the environment, and has a competence independent from the other two components. This independence means that the reaction component does not depend on the existence of a plan to act. The projection component explores possible futures and compiles appropriate reactions. These compiled reactions are expressed as situation-action rules which we call Situated Control Rules (SCRs). When available, these SCRs are given preference by the reaction component during action selection.

Our projection process considers events under the system's control and external events caused by the environment or other agents. Projection uses a domain causal theory represented as a "plan net." Uninformed projection is simply a search through the space of possible event sequences allowed by the plan net and, thus, is infeasible in realistic domains. To achieve efficiency, projection should be controlled. We are considering two ways to control projection: first, by projecting actions selected with reference to the systems overall behavioral constraints; and, second, by limiting the projection of external events with a model of the probability of event occurrence.

Behavioral constraints are expressed in a language based on branching temporal logic. In this language it is possible to express constraints of achievement, maintenance, and prevention. Unfortunately, these constraints are often not in a form which can usefully control the temporal projection search. We are using the third system component, problem reduction, to translate behavioral constraints into search advice for the temporal projection. The problem reduction component, REAPPR, uses domain and problem specific planning expertise to recursively decompose problems into appropriate subproblems. This search through the reduction space can provide guidance to the projection component. In the way that projection informs reaction, so does reduction inform projection.

The eventual goal of the ERE project is a set of software tools for designing and deploying integrated planning and scheduling systems that are able to effectively control their environments. To produce such software tools, we are working towards a better understanding of the theoretical aspects of action selection and sequencing in terms of action execution. Work in this project thus involves both theory and implementation. We are working with others to define a set of benchmark problems and evaluation metrics. With these benchmarks and metrics we will be able to more objectively and comparatively analyze the performance of our architecture. We plan to implement the benchmark problems in software simulations and in a hardware test bed.

Current theoretical work in the group addresses the problem of when to plan and when to act; the integration of problem reduction with temporal projection; probabilistically controlled filtered beam search with anytime properties; and closed-loop hierarchical control systems.

Current implementational work in the group revolves around a set of reactive planning and control problems grounded in a Sun and X11 based simulation called the Reactive Tile World. We are currently extending the existing REAPPR problem reduction system and integrating it with existing temporal projection code.

Milestones:

1990: • Develop a set of benchmark tasks and evaluation metrics.
• Design a hardware test suite to support benchmark problems.
• Design alternative simulation test environments.
• Design and implement problem reduction and temporal projection interfaces.
1991: • Install a simple hardware environment to test architecture.
• Evaluate problem reduction and temporal projection interaction.
1992: • Complete performance analysis of plan nets and situated control rules with problem reduction.
• Facilitate use of benchmarks and metrics by other research groups.
1993: • Extend architecture to deal with goals of information receipt.
• Design and implement interface to problem reduction and temporal projection system.
1994: • Apply architecture to other NASA domains (Lunar habitat, space station, etc.).

Planning and Reactive Control

Nils Nilsson; Stanford University
Stan Rosenschein, Leslie Kaelbling; Teleos Research

This project is focussed upon the development of intelligent reasoning systems that are situated in real-world, dynamic environments. These systems must react to unexpected events and conditions that occur due to their own actions as well as the actions caused by external agents.

The 1990 work at Stanford consists of a number of inter-related efforts. Some of these efforts are parts of Stanford PhD dissertations, and some are the work of Professor Nilsson himself.

Professor Nilsson has been working on problems of reactive planning and plan execution, and on "action networks" and an action-network language. In the area of reactive planning, the concept of "tree-plans" has been explored. These structures are sub-trees of a state-space spanning tree rooted at the goal node. They can be represented conveniently as "extended triangle tables." Tree plans have much more built-in conditionality than do conventional plans. This conditionality is important in applications in which one cannot accurately predict the consequences of actions as well as in situations in which other agents are also performing actions. Tree plans, in extended-triangle-table form, can be executed by using an extension of the triangle-table scanning algorithm. This algorithm has been implemented and tested on some simple examples.

Action networks are networks of logical gates that are used for computing and selecting actions for agents. They are particularly useful in dynamic environments--such as those in which several agents are performing tasks. A computer language for specifying action networks (called ACTNET) has been developed. Execution of a program in ACTNET causes an action network to be built. ACTNET supports parameter binding and recursion. Parameter binding allows the same program to construct different action networks depending on the arguments passed to the program upon execution. Recursion allows the construction of an iterated network structure to solve complex tasks. We have tested the ACTNET language on a variety of simple simulated tasks and are preparing to try it for controlling real robots at Stanford.

PhD dissertation work involves a number of basic research topics including: "universal attachments" for linking programs and data structures to expressions in declarative representations; a hierarchical planner that generates its own hierarchies; agent negotiation; and, a theory of explanations. The "universal attachment" work is intended to augment existing logical reasoning techniques with procedural code. This approach rigorously maps procedural code onto formal systems. The hierarchical planning work concentrates on abstraction. The goal of this work is to develop planning techniques that exploit inherent abstractions in a domain to support efficient search. The agent negotiation work concentrates on multi-agent coordination. The goal of this effort is to develop a framework for negotiation within which agents can communicate to achieve their sometimes conflicting goals. Finally, the explanation effort concentrates on the development of a generic explanation theory that could be used by any reasoning system to explain its conclusions.

1990 research at Teleos is concerned with the integration of deliberative reasoning about action with real-time reactive control using the situated automata approach to agent architecture. The situated automata approach is based on the idea that the specification of an agent's behavior can rely on symbolic representations, but to achieve real-time behavior, this specification should be compiled into a circuit-like representation that does not perform conventional symbolic processing. Various design tools, including Rex and Gapps, have already been produced that allow a programmer to specify behavior declaratively and automatically generate a circuit with real-time execution bounds. The reactive behavior of the circuit is achieved using a bounded computation to update the agent's model of the environment and to compute the next action to be taken. While reactive behavior is absolutely crucial for real-world agents, it is not sufficient in itself. When resources are available, the agent should reason about its possible future actions before acting. This research activity concentrates on adding a planning capability to the situated-automata architecture. In 1991 and beyond, Teleos intends to extend the approach and apply it to the problem of acting to gain information. Applications of this type require the active control of sensors and sensory processing, as well as the interleaving of perceptual actions with actions of other types.

The program is also committed to testing and evaluating various architectures empirically on a robotic testbed. A new activity, to begin in March, 1990 with DARPA co-funding, will carry out this empirical evaluation by developing a set of benchmark tasks and evaluation metrics for integrated agent architectures and by conducting rigorous experiments on the testbed.

In 1990, Teleos has continued to investigate the integration of deliberative planning and reactive control. In addition, Teleos has designed and implemented a skeleton system, based on the situated-automata architecture, to support experiments in real-time perception, planning, and reactive control. An important feature of this system is a dynamic database containing descriptions of objects in the environment. This database is continuously updated as a function of the stream of sensory inputs and makes explicit the uncertainty that the agent has about its environment. Preliminary work was done on integrating real-time visual processing with the approach to reactive control embodied in the Rex/Gapps programming system.

In 1991, Teleos will continue its program of research on planning and reactive control, with special emphasis on actions designed to gain information, as well as on rigorous experimental validation of architectural concepts. Specifically, the following activities will be undertaken: Rex/Gapps will be extended to incorporate more sophisticated run-time planning, including planning to gain perceptual information; Teleos will develop an experimental testbed, in collaboration with Ames personnel, to support evaluation of agent architectures; and, experiments will be carried out that focus on real-time sensory-guided robotic manipulation tasks.

Milestones:

1990: • Implementation of planning component within the situated-automata suite of tools.
1991: • Specification of benchmark tasks and evaluation metrics.
• Implementation of information-gathering strategies using Rex/Gapps and real-time perception modules.
1992: • Integration of situated-automata/planning with robotic hardware testbed.
1993: • Carry out performance analysis of situated-automata architecture on benchmark tasks.
1994: • Demonstrate robotic system combining planning and reactive control in complex tasks requiring sophisticated perception and goal-directed action.

Machine Learning and Planning in Dynamic Environments
Jaime G. Carbonell, Matthew Mason, Tom Mitchell; Carnegie-Mellon University

This research effort addresses the design and implementation of autonomous agents that integrate aspects of sensing, planning, execution, and learning. One project focuses on developing a meta-reasoning architecture that lets one

explicitly reason about different planning methods, experimentation strategies, and learning techniques. The other explores learning in the context of robotic manipulation tasks that involve sensing, reacting, and planning. There is particular need for such systems in various NASA programs, such as subsystems for the space station, planetary and space exploration, and manned space platforms.

Carbonell and Kuokka have developed MAX, an integrated architecture that incorporates many ideas from his work with Minton on Prodigy, but that also supports reasoning about meta-level actions. As a result, the system can reason about when to plan vs. execute a canned procedure, when to modify a plan vs. replan, and when to experiment vs. apply existing knowledge. Upon making such decisions, MAX uses an explanation-based learning method (like that in Prodigy) to store a generalized rule that bypasses the need for future reasoning in similar cases. Carbonell and Kuokka have implemented an initial version of the architecture and tested it in the domain of robotic path planning in environments involving both predictable and unpredictable change.

In related work, Carbonell and Gil have developed methods for systematic experimentation that can be used to extend an incomplete domain theory. This approach notices when some plan fails to achieve a desired goal, and then attempts to determine the source of the problem by varying the nature of the objects involved or by varying parameters on the actions (e.g., the magnitude of an applied force). Experimentation may reveal that an existing operator description is incomplete, in which case the system adds some new condition or action to improve its domain theory (and thus it planning ability). The researchers have tested this system on a variety of tasks, including production of optical-telescope mirrors and exploration of unknown terrains.

In another project, Mitchell and Mason have used a hand-eye testbed to explore learning in robot manipulation. The basic task involves tilting a tray so as to slide an object to a desired location. Despite its apparent simplicity, this task has proven challenging to the robotics community, making learning an attractive approach to acquiring the necessary domain knowledge. Mitchell and Mason have constructed a number of simple learning agents that operate in this testbed, and that employ different amounts of background knowledge in the learning process. Recent results have shown that these agents can learn effective manipulation strategies that produce 95% success rates at positioning objects at randomly selected target positions and orientations. Furthermore, such learning techniques are effective across a variety of irregular object shapes and in the presence of complicating factors, such as additional invisible objects in the tray. These results represent one of the first robotic tasks for which learning techniques significantly outperform state-of-the-art analytical planning methods.

During 1990, the team will complete implementation of the MAX meta-level architecture, adding knowledge about alternative planning methods, about approaches to repairing incorrect plans, and about various learning methods. They will also carry out a systematic evaluation on the experimentation methods

to determine the extent to which they let the system recover from incomplete domain knowledge. Finally, additional experiments will be run in the hand-eye testbed, focusing on different learning agents, more complex object shapes, and the ability to generalize to different situations.

In 1991, the project will begin the redesign of the MAX architecture to allow its incorporation into Prodigy, thus enabling meta-reasoning to consider the full range of Prodigy's planning methods (case-based, planning, linear planning, complete non-linear planning, hierarchical abstraction planning) and learning methods (experimentation, explanation-based learning, static reformulation, analogical transfer, formulation of abstraction hierarchies). It will also extend the approach beyond tray tilting to the task of pushing blocks and then to the more complex task of grasping objects.

Milestones:

1990: • Complete implementation of the MAX meta-level architecture.
• Demonstrate experiments of increasing complexity in the hand-eye testbed.
1991: • Complete new design of MAX architecture.
• Demonstrate pushing and grasping tasks in the hand-eye testbed.
1992: • Select the best learning system from previous studies with the hand-eye testbed and begin more detailed evaluation on complex manipulation tasks.
1993: • Complete of evaluation of hand-eye learning system and initial incorporation into a teleoperated manipulation device, allowing a continuum from remote to autonomous control.

Adaptive Planning
Smadar Kedar; RIA

The long-term objective of this research is to augment reactive systems (systems that react to dynamic environments) with the ability to refine their interaction with the world through learning from experience. In particular, we are examining situations in which the system may fail to react appropriately, and would learn from its mistakes.

For future NASA missions, the ability to autonomously react and refine reactions through experience will be needed when human teleoperation of a robot may not be possible. Such scenarios may include teleoperation when the time delay is too long (e.g. unloading payloads from a descent vehicle on Mars), or when low-level actions are difficult for humans to control (due to vibrations or unpredictable movements). In such scenarios, teleoperation commands may need to be more high-level, while fine-grained actions would be generated and refined more autonomously.

The research motivations for this work are two-fold. First, most current reactive systems can only react to situations which have been completely specified *a*

priori. For unanticipated conditions, these systems may fail to react at all. More robust reactive systems need to be augmented with capabilities for detecting such failures, repairing them, and augmenting their reactive rules in a general way so as to learn to avoid such failures in the future. Second, most symbolic machine learning approaches to learning from failure are limited in that they do not work in reactive situations, but assume a "plan-then-execute" model of action. These approaches need to be applied and tested in reactive systems.

Our approach is to focus the research to the domain of robotics control, in particular, hand-eye coordination. We are collaborating closely with personnel involved in the task described above, in particular with Leslie Kaelbling. The testbed for our research (located at Teleos Research) is a Zebra ZERO force-control robot arm and a 2D vision system, which communicate through an ethernet to a workstation. Software tools include the Rex and Gapps high-level robotics programming languages.

In 1990, we have focused our initial efforts on problems of failing to react in a calibration and tracking scenario. (To calibrate vision with arm movement, for any point in the robot arm coordinate frame, a corresponding point in the vision coordinate frame is found. Once they are calibrated, the robot arm tracks an object across the visual field, and poses above it.) Given an initial set of reactive rules for calibration and tracking, certain exceptional conditions (such as additional objects other than the arm in the visual field during calibration, or a loose gripper) are not accounted for.

To address this problem, we propose a novel reactive system architecture which, along with its reactive rules, has a list of possible error conditions for the arm, vision, and reasoning systems (e.g. lose gripper). As failures happen, a strategy to recover from failure is invoked. In parallel, the cause of the failure is explained using the theory of possible errors (or through an explanation from the human), and then generalized applying explanation-based learning from failure. The generalized conditions of the failure are "compiled-as-needed" into the original set of reactive rules if it is believed that the failure will recur. As an alternative to the automatic modification of the rules, the system can be used as a tool for a user to experiment and refine reactive rules based on suggestions from the system.

The bulk of our work in 1991 will be on testing this architecture, first in software simulation, and then on the hardware testbed. We will also begin to explore its application to a NASA domain.

Milestones:

1990: • Augment the Gapps interpreter/compiler to support learning from failure.
• Design and implement the learning component in the scenario.
1991: • Port system to and test on the hardware testbed.
• Apply to NASA domain (e.g. teleoperation of a planetary

Rover).

1992: • Augment the learning system by enabling the automatic generation of reactive rules.
• Demonstrate on a suitable NASA application.

4.5 Intelligent Control

Intelligent control is a new paradigm for bridging the symbolic techniques of AI with the analytical schemes of control theory. Intelligent control can remove many of the limitations of the conventional analytical methods of designing control systems (e.g., the modeling of very complex processes). In intelligent control, knowledge of the control process plays a significant role and approximate reasoning methods based on the theory of fuzzy sets and neural networks can provide appropriate schemes for representing and reasoning with much of the qualitative knowledge used in control.

Fuzzy Control

Lotfi Zadeh; University of California at Berkeley

The major goal of this research is to develop computationally-effective methods for the design and analysis of knowledge-based control systems which operate in an environment of uncertainty and imprecision. In 1990, the work is focused on two application areas: replacement of a skilled human operator by a fuzzy rule-based system; and replacement of an expert by a fuzzy Prolog-based system which has the capability to draw conclusions from facts and/or rules which are probability-qualified, with probabilities expressed as values of linguistic variables. In the latter area, an important role is played by dispositional logic, a logic in which the propositions are assumed to be proponderantly but not necessarily always true. Dispositional logic provides an alternative method for dealing with commonsense reasoning and the problem of exceptions.

A recently initiated direction in our research relates to what might be called "interpolative reasoning." Interpolative reasoning is related to both analogical reasoning and case-based reasoning and exploits the capability of the rules of inference in fuzzy logic to apply to premises which do not match exactly any of the antecedents in the rule-base. Interpolative reasoning plays an essential role in fuzzy control as well as in qualitative systems analysis.

A promising area of research initiated by a doctoral student, C.C. Lee, in collaboration with Dr. Hamid Berenji of RIA, involves the design of fuzzy-neural systems in which the fuzzy rules in the rule-base are learned or tuned through the use of neural network techniques. The validity of the proposed method was tested by an application to the balancing of an inverted pendulum, and the results of simulation studies have shown that fuzzy-neural techniques provide an effective approach to the design of self-learning and/or adaptive systems.

Our research has a direct bearing on the analysis and design of systems which are employed directly or indirectly in space missions of various types. In particular, the methods under development provide a basis for improving the robustness of critical components of complex systems and reducing power requirements.

In 1991, we plan to continue development of the techniques described above and, in addition, investigate the following problem areas:

1. Inference of fuzzy if-then rules from observed data. In particular, we plan to explore the possibility of identifying a collection of rules which represent a dynamic rather than a static system.

2. Development of a better understanding of issues relating to the stability of fuzzy control systems.

3. Development of general methods for the optimization of fuzzy rule-based systems through the use of neural network techniques.

Milestones:

1990: • Develop dispositional logic.
1991: • Develop methods of interpolative reasoning and prepare a report for publication.
1991: • Apply interpolative reasoning to propagation of belief with linguistic probabilities.
1992: • Develop a general technique for the adaptation of fuzzy systems by using neural network methods.
1993: • Develop an inference engine for uncertain knowledge management based on the solution of relational equation rather than a rule-based system.
1994: • Develop a knowledge representation system based on dispositional logic for exception handling and interpolative reasoning.

Approximate Reasoning
Hamid Berenji; RIA

A major difficulty in the design of intelligent controllers based on Approximate Reasoning and Fuzzy Logic is how to fine-tune the control rules. These controllers have been shown to be viable alternatives to analytical controllers especially when complete mathematical models are not available (e.g., in many non-linear dynamic systems). The goal of the current research is to integrate unsupervised learning into the design of approximate reasoning-based controllers which will automate the fine-tuning of control rules. The approach selected here is to start with small but challenging control problems (e.g., the cart-pole balancing problem) and demonstrate the feasibility of using the hybrid approximate reasoning and reinforcement learning model. We will then extend the approach to larger size, challenging control problems (e.g., trajectory

generation for a three-joint robotic arm). This research will directly contribute to the design of controllers in NASA domains such as process control and in-space assembly tasks.

In 1990, we have completed the design and development of a prototype cart-pole balancing system based on two types of control: approximate reasoning (i.e., related to fuzzy logic based control) and conventional control (i.e., State Feedback Control). The hybrid system performs substantially better on standard benchmark control problems than the conventional system alone. We also developed a model for a controller based on approximate reasoning that combines rule-based control with neuron-like elements that learn by improving predictions over time.

Our plan in 1991 centers on three main topics: the application of our reasoning-based control algorithms to a larger scale problem; the extension of approximate control theory; and, the continued development of Team Theory. We intend to apply the above control technique to a larger scale control problem such as trajectory control of three-joint robotic arm. This will test the robustness of the control algorithms as well as their generality. The second topic involves the development of new combination operators for approximate control. In particular, our initial study of the recently introduced Ordered Weighted Average (OWA) operators by Yager has produced interesting results and we plan to present the results of this study to the 29th IEEE Conference on Decision and Control (CDC-90). Lastly, we intend to publish the results of joint research with Professor Thomas Whalen on extending epistemic logic to include reasoning about actions from the perspective of Team Theory (developed by Marshack).

Milestones:

1991: • Apply approximate reasoning-based control to a three-joint robotic arm.
1992: • Develop a general theory for approximate reasoning based on reinforcement learning and neural nets.
• Prepare a publication (journal paper) based on this work.
1993: • Study the application of the above theory in a selected NASA domain.
1994: • Complete the development of a general integrated approach for uncertainty management in AI systems.

4.6 Machine Learning for System Maintenance and Improvement

The complexity of advanced software used in NASA projects generally necessitates long and expensive software development times. Furthermore, the resulting systems may contain errors. One of the most promising areas for artificial intelligence is the application of machine learning techniques to aid and simplify the software development process. Specifically, it may be much easier for a software engineer to create a simple version of his program that is

incomplete or inefficient, and then use automatic learning techniques to refine and improve the initial program, as compared to creating the finished program from scratch. The following research projects are directed towards this goal, and they span the continuum from very applied, comparatively short-term projects, to longer-term projects that are concerned with very basic issues.

Learning and Performance Improvement in Scheduling
Steve Minton, Andy Philips; RIA

The major goal of this work is to integrate machine learning methods with scheduling systems in order to develop schedulers that improve their performance over time. This involves two areas of research: the design of a framework for scheduling heuristics and the development of mechanisms for automatically learning such heuristics during operation of a scheduler.

In 1990, we are analyzing a scheduling heuristic which is based on our studies of the Guarded Discrete Stochastic (GDS) network. This network was developed by researchers at the Space Telescope Science Institute for scheduling the Hubble Space Telescope. Our heuristic guides the scheduling process by minimizing the total number of constraint violations in a schedule. We have shown that the heuristic produces very good results (improving on the GDS network) in some standard test problems, and we are now attempting to generalize the heuristic and try it on other problems.

We are also studying the use of Explanation-Based Learning (EBL) in conjunction with such scheduling heuristics. EBL is an analytical learning technique in which a system proves that a specific instance is a member of a more general class. We call such a proof the explanation of the instance. The system then derives the weakest conditions under which the explanation holds. If these weakest conditions hold then the object is guaranteed to be a member of the general concept. Systems that perform this process are learning concise and general concept descriptions that are usable later without explanation. In order for an EBL method to be useful in a practical application, it must be used in conjunction with an efficient performance system. Furthermore, previous work with the MORRIS system and the PRODIGY system has shown that the learning method must be sensitive to the heuristics used by the performance system. Therefore, we expect our analysis of our scheduling heuristic to be useful not only in its own right, but also when EBL is applied to the scheduling system.

In 1991, we will continue work in on scheduling heuristics and methods for automatically learning and improving those heuristics. In particular, we will develop systematic learning methods to replace current ad-hoc methods. We will also begin to study the relationship of linear and nonlinear planning. Nonlinear planning, developed in the mid-1970's, was described as an improvement over the classical linear planning approach. Non-linear planning has become widely used, and the claim that it is better than linear planning is almost universally believed. However, our studies show that there are a set of

complex tradeoffs that one makes when one approach or the other is adopted. We are developing formal models of linear and nonlinear planning in order to be able to compare the relative efficiencies of these techniques.

Milestones:

1990: • Demonstrate applications for new scheduling heuristic.
• Publish conference paper on scheduling heuristic.
• Develop initial EBL scheduling.
1991: • Develop advanced learning techniques for systematic search.
• Publish conference paper on nonlinear planning.
1992: • Apply advanced learning techniques for systematic search to real scheduling problems.
1993: • Develop advanced learning techniques for iterative improvement search.

Icarus - An Integrated Architecture for Learning

Pat Langley, Kevin Thompson, Kate McKusick, John Allen; RIA

The Icarus project is developing a conceptual framework for the control of autonomous intelligent agents. The goal is a single system that covers a broad range of cognitive behavior, including recognition of complex physical objects, generation of plans, and execution of motor skills. The eventual aim is to integrate these aspects of cognition into an agent that can interact with an unpredictable environment and acquire knowledge from its experience.

The framework assumes a single underlying organization for long-term memory--a hierarchy of probabilistic concepts--to store three types of knowledge: physical object concepts, plan knowledge, and motor schemas. Using a process of heuristic classification, the agent adds new experiences to the hierarchy, which effectively organizes knowledge in long-term memory. Intelligent action emerges not only from large amounts of domain knowledge, but also from the ability to efficiently find the "best" knowledge for a given goal and situation. The working hypotheses of Icarus are that a single long-term memory can represent many kinds of knowledge, that heuristic classification can retrieve this knowledge effectively, and that a single learning mechanism--concept formation--is sufficient to incrementally acquire knowledge from experience.

Icarus consists of three major components: Labyrinth, Daedalus, and Maeander. The Labyrinth system classifies complex physical objects and stores abstractions of them in long-term memory. The agent can use this stored knowledge to recognize similar objects and to guide its interaction with an object it may need to manipulate or avoid. The Daedalus system generates plans and acquires plan knowledge by classifying and storing successful plans in long-term memory. In drafting plans, the agent generates sequences of actions that it must follow in order to achieve its goals. By storing successful plans in the long-term memory, the agent learns appropriate responses to

situations similar to those it has experienced. The Maeander system acquires motor schemas, which the agent can use to recognize a familiar action or to execute an action included in a plan. These three components are currently separate, but integration into a single architecture is a research priority.

In 1990, Labyrinth has been extended to handle objects with many components and to support relations among these components. Initial tests on a diagnostic task and on artificial data have been encouraging. In addition, we have developed an initial version of Daedalus and are currently testing the system's planning ability. An initial version of Maeander has been built and tests have begun using a simulated robot arm. An initial user interface for Labyrinth have been constructed; this will be generalized to other components of Icarus.

In 1991, we will complete the integration of the three Icarus components into a functional system. We will conduct extensive testing on simulated and actual (although initially simple) task domains in order to empirically study system performance. We will also begin an investigation on mechanisms for resolving goal conflicts and for dealing with external interrupts.

A long-term goal is for Icarus to control an autonomous agent that can function in unpredictable and unfamiliar environments, much as a human would. Potential applications include NASA missions in which an autonomous agent might function in lieu of a human astronaut, as in planetary exploration, the execution of deep-space scientific experiments, or Space Station assembly and maintenance. In fact, such an agent would be suited to any hostile environment where it would be costly or hazardous to send humans: to do machine diagnosis/repair or substance cleanup in radioactive areas, or as a "trouble-shooting" rover in support of a martian or lunar habitat.

Milestones:

1990: • Complete initial evaluation of Daedalus and Labyrinth systems.
- Modify Daedalus to employ Labyrinth in storing and retrieving plan knowledge.
- Complete initial tests of the Maeander component.

1991: • Extend Maeander to use Labyrinth for storing and retrieving motor knowledge.
- Modify Daedalus to employ Maeander for execution of actions.
- Incorporate mechanisms for resolving goal conflicts.

1992: • Begin Icarus evaluation in an autonomous agent testbed.

1993: • Complete evaluation of the Icarus architecture in autonomous agent testbed and begin adaptation to planetary rover or teleoperated device.

1994: • Incorporate Icarus software into a prototype rover or teleoperated device; begin initial field tests.

SOAR Architecture

Paul Rosenbloom; USC-Information Sciences Institute

SOAR is a general cognitive architecture that integrates problem solving and learning. The system represents knowledge as production rules that specify preferences for certain states, operators, or goals, and it uses this knowledge to direct search through various problem spaces. Before making a decision about which state to expand or operator to apply, SOAR's rules add elaborations and preferences to working memory, which it then uses to select a state or operator. The architecture acquires knowledge through "chunking," a form of explanation-based learning that caches the results of previously solved problems. Professor Rosenbloom's previous and continuing contribution to the overall SOAR effort has focused on the issues of knowledge-level learning, expensive chunks, and combining analogical and rule-based reasoning.

During 1990, there have been four major developments in the effort on knowledge-level learning: (1) the creation of a simplified memorization operator which takes as input two arbitrary graph structures in working memory, and which acquires a chunk that encodes the pair; (2) the extension of this capability to carry out induction from multiple examples, thus providing the first compelling demonstration of nondeductive knowledge-level learning; (3) the development of a new induction algorithm -- implemented in SOAR--which (under restricted circumstances) learns the same concepts as Mitchell's candidate elimination algorithm, but with polynomially-bounded time and space; and (4) the extension of this induction capability to utilize simple forms of explanations, determinations, and irrelevance knowledge.

In the area of expensive chunks, Rosenbloom has developed a framework for production matchers that are bounded in space and time. The basic approach involves restricting the expressibility allowed in productions and working-memory elements, and limiting the degree of consistency that the match guarantees among the bindings of different variables. One new result is the optimality (under certain conditions) of unique attributes (a specific representational scheme) along the trade-off between expressibility and efficiency. Other results include the design of several novel production matchers and a preliminary analysis of three new match algorithms based on tree-structured productions. This work has implications for any NASA application that represents domain knowledge in production rules.

In the area of analogy, implementation is nearly complete for a system that pronounces proper names. This system combines rules and analogy by using rules to suggest default pronunciations and using compelling analogies to override inappropriate rule applications. It also employs background knowledge about punctuation, morphology, syllabification, and language of origin to produce good phonetic transcriptions and stress assignments. Finally, it uses a partial theory to abductively reconstruct the justifications behind given examplars -- and to extend that theory when it cannot explain the exemplar -- so that it can later be used in analogies.

Milestones:

1990: • In knowledge-level learning, investigate a new, simplified approach to the data-chunking problem--using explanation-based learning to memorize information in such a way that it can be retrieved without already having it available at retrieval time -- based on a limited form of partial match.
• For expensive chunks, complete the analyses of unique attributes and the three tree-structured matchers, and examine at least one non-tree-structured alternative with the new match algorithm.
1991: • Complete the analogy effort by finalizing system implementation and by running a set of experiments evaluating its efficacy and its dependence on various forms of knowledge and processing.
1992: • Continue refinement of the SOAR matcher and chunking mechanism, testing the architecture on diagnostic or scheduling tasks.
1993: • Reimplement a fielded NASA system in the extended SOAR framework.
1994: • Carry out systematic comparisons between existing NASA system and the SOAR implementation.

SOAR and the External Environment

John E. Laird; University of Michigan

The research on SOAR at the University of Michigan is directed at studying the problems that arise when an integrated learning and problem-solving system interacts with an external environment. To this end, Professor Laird and his colleagues have developed on a major new version of SOAR that supports interaction with an external environment. In addition to having a general input/output interface, SOAR's operators now destructively modify a state instead of creating a new state. A new release of SOAR 5 is now available, along with a major rewrite of the SOAR manual. This augmented formalism has been tested in three major areas: visual attention, robotic control, and learning from the external environment.

For SOAR to interact with an environment, it must be tightly integrated with its perceptual system. One key idea is the use of visual attention as the mediator between low-level vision and high-level cognitive control. Visual attention is used as a method of limiting and localizing information within the visual field. The core of the model is a collection of operators that are sufficient for fixed-eye visual tasks in two dimensions. The model has been successfully used to account for psychological data on visual precuing and search experiments to an accuracy of fifty msec. Since it involves a computer implementation, this work also has potential applications as a tool for computer vision and for the design of visual displays. For example, it could lead to improved vision systems for satellite reconnaissance, as well as for planetary exploration. It could also lead to better interfaces for use in mission control.

The most successful project in interfacing SOAR with external environments has been Robo-SOAR, which has linked SOAR to a Puma robotic arm and an accompanying vision system. The arm has a special controller that eliminates the need for SOAR to compute joint angles and similar details. Laird has used this testbed to examine issues of reactive planning and execution, in which one is always reevaluating decisions in light of changes in the environment. If Robo-SOAR has previously encountered the current situation, either through internal planning or previous interactions with the environment, it responds immediately. Otherwise, the system falls back on internal problem solving. (Details of this work were reported at the NASA Conference on Space Tele-robotics in January, 1989.) Laird is also examining the issues that arise when one must interrupt work on a problem because a higher-priority problem unexpectedly occurs. Such interruptions can affect learning on the initial task, and they also require the agent to return to the original task after solving the high-priority problem. Preliminary studies have demonstrated Robo-SOAR's ability to respond quickly to interruptions and changes in the environment.

In 1990, Laird and his colleagues have connected SOAR 5 to a Hero 2000 mobile robot, which has a simple interface (it was connected to SOAR within two days) and a variety of sensors. The resulting system, Hero-SOAR, is now able (in real time) to search its environment for cups, pick them up, and, after finding a box, drop the cups in the box. One problem with the Hero is that it often loses commands, and its responses are often garbled. Hero-SOAR deals with this issue by waiting after a command, then re-reading sensors and, if necessary, re-executing the command to achieve its desired goal. Ideas from both Robo-SOAR and Hero-SOAR could lead to improved methods for controlling autonomous vehicles for planetary and space exploration.

In addition, Laird has demonstrated SOAR's ability to learn new tasks from interacting with a human, either through a set of instructions or through careful tutoring. This has proven successful in both Robo-SOAR and in a series of experiments on toy tasks where SOAR learns a new task through instruction, such as Tic-Tac-Toe and Block's World. However, this approach relies heavily on instruction, and recent work on learning from the environment has focused on more autonomous methods. Here the system has basic capabilities for interacting with its environment, but it has no internal model of how its actions affect the world. Without such a model, it can not plan its actions but instead must rely on reactive behavior. In order to acquire a model, SOAR must perform actions in the environment, observe their effects, and induce the cause-effect relationships between them. Laird has implemented an initial system along these lines and has tested it for two simple domains, in which the relevant effects are close to the locus of actions in space and time, and in which the space of possible features is small. This work has implications for planetary exploration, in that programmers cannot fully anticipate the effect of actions on an autonomous agent's environment.

Milestones:

1990: • Expand the model of visual attention to account for a wider range

of experimental data, including illusory conjunctions, global precedence, and visual persistence.
- Extend the work on unsupervised learning from external environments to domains in which behavior is conditional on environmental features, to more complex actions where one must consider time and spatial extent, and to hierarchical actions.
- Implement the resulting methods in both Hero-SOAR and Robo-SOAR.

1991:
- Use the attention model as a basis for a SOAR robotic vision system that can learn to compose visual operators to develop routines for new goal-directed visual tasks.
- Augment SOAR to include an internal model of time, and test the resulting system on tasks which involve alternative actions that vary in the time they take to execute and in their reliability.
- Examine both problem solving and learning in this context of SOAR improvements.

1992:
- Develop an integrated hand-eye system based on SOAR, and begin evaluation of the system on manipulation tasks.

1993:
- Complete evaluation of SOAR hand-eye system and begin incorporation into a teleoperated manipulation device, allowing a continuum from remote to autonomous control.

1994:
- Incorporate the SOAR hand-eye system into an in-flight manipulator.

Improving Search-Based AI Systems

Thomas G. Dietterich; Oregon State University

The central issues addressed in this research project are the discovery of new abstract objects and operators and their use in improving the efficiency of problem solving. The focus is on solving optimization problems such as occur in minimax search, scheduling, and engineering design. Problem solving in such domains can be viewed as search for some state that satisfies a set of constraints, and these constraints can be used to direct the learning process.

Initial work by Flann and Dietterich has focused on chess endgames. They have developed an abstract domain theory that lets them apply explanation-based learning techniques to extract general sufficient conditions for achieving a given goal (say, avoiding checkmate) from search trees for two-person games. They use these sufficient conditions to define new abstract objects and operators that ignore irrelevant details and thus lead to more rapid problem solving. They have also demonstrated a method for compiling these concepts into extremely fast recognition rules by exploiting geometrical knowledge of the domain. The result is a speedup in problem-solving efficiency of more than five orders of magnitude. This improvement is significantly greater than that observed in previous work on learning in problem solving.

Related research by Cerbone and Dietterich has addressed the task of two-dimensional structural optimization in engineering design. In this case, they have developed a technique for discovering optimal design rules from successful designs, which they are now implementing. The approach involves

first solving a set of simple problems using traditional numerical optimization techniques. The solutions are then analyzed to discover relationships that let the problem-solving system construct the optimal solutions directly from the given problem parameters. The analytic process employs abductive reasoning, a method that constructs explanations that include default assumptions. Hand simulations of this technique have produced optimization rules that were previously unknown to the researchers.

In 1991, this project will focus on integrating the methods developed in 1990 with a NASA scheduling system. This will then be used to guide further research into the automatic improvement of search-dependent problem solving.

Milestones:

1990: • Extend Flann's technique to handle all of Quinlan's databases for chess endgames.
• Generalize the technique so that it can be applied to other domains.
• Implement and test Cerbone's method in 2-D structural design.
1991: • Implement and demonstrate the use of search improvement techniques on a NASA scheduling problem.
1992: • Begin systematic evaluation of a learning scheduler and incorporate the results into an improved system.
1993: • Continue evaluation and improvement of learning scheduler and begin adapting the system for field tests.
1994: • Application of learning scheduler to actual NASA scheduling tasks and comparison to existing fielded systems.

Utility and Incomplete Theories in Explanation-Based Learning
Raymond Mooney; University of Texas, Austin

This project focuses on three issues in explanation-based approaches to learning. The first is the utility problem, which involves ensuring that learned rules actually improve problem-solving efficiency rather than degrade it, as has occurred in some recent systems. The second is that most explanation-based learning methods rely on the presence of a complete domain theory and have no way to extend the domain knowledge they are given at the outset. Finally, most methods for explanation-based learning rely on deductive reasoning techniques that are unrealistic in many domains, rather than abductive reasoning methods.

With respect to the utility problem, Mooney has developed methods for limiting the use of learned rules and has conducted experiments which demonstrate that these methods ensure improved performance. Specifically, he has identified three techniques for improving the utility of rules acquired through explanation-based learning: limiting chaining on learned rules; avoiding the use of learned rules to solve subgoals when it interacts with other subgoals; and, using depth-first iterative deepening search control.

With regard to incomplete domain theories, Mooney has developed Induction Over the Unexplained (IOU), a approach to learning that combines existing empirical and analytical techniques. IOU uses explanation-based methods to learn part of a concept and uses inductive methods over unexplained aspects of examples to impose additional constraints on the final concept definition. He has conducted experiments in two financial domains (classifying stocks and predicting bankruptcy) that demonstrate IOU's ability to use incomplete theories to learn more accurate concepts from fewer examples than a purely empirical method like Quinlan's ID3. He has also used methods from learnability theory to formally prove that IOU can learn from fewer examples. Finally, he has used IOU to model psychological data demonstrating the effect of background theories on human concept acquisition.

Finally, Mooney has developed an AI system for abductive reasoning which constructs explanations that require default assumptions. His ACCEL system uses a metric for explanatory coherence in the construction and evaluation of these abductive explanations. He has used several examples to show that the evaluation metric, which measures how well an explanation connects a set of observations, is superior to standard simplicity metrics at choosing the best explanation. ACCEL uses this metric as a heuristic for search during backward chaining to ensure the efficient construction of high-quality explanations.

Mooney has tested the system on a number of examples in the domains of explaining animal coloration and human intentional behavior. However, the work also has applications to the diagnosis of complex physical devices, such as the control systems for space stations and spacecraft. He has also run experiments demonstrating that ACCEL's heuristic search method greatly reduces run time while still achieving optimal explanations in terms of the coherence metric.

The focus of research in 1991 will be on formal evaluation of IOU and extension of ACCEL to the construction of explanations in a NASA physical system domain. We are now evaluating whether the diagnosis of faults in a thermal control system (the SADP TCS domain) is appropriate.

Milestones:

1990: • Develop next version of IOU that can both learn new rules and modify or remove existing rules.
• Refine the heuristic search procedure used in ACCEL to further increase efficiency.

1991: • Evaluate IOU both empirically and by means of formal analysis.
• Evaluate the ACCEL system's ability to efficiently construct correct explanations in a NASA domain, possibly diagnosis of faults in a thermal control system.

1992: • Continue evaluation and improvement of ACCEL's diagnostic ability, incorporating ideas from the IOU system to acquire knowledge for diagnosis.

1993: • Begin adapting the combined ACCEL/IOU system to a specific NASA diagnostic problem.
1994: • Apply the ACCEL/IOU system to a NASA diagnostic task and compare its behavior to existing fielded systems.

Representation in Incremental Learning
Paul Utgoff; University of Massachusetts, Amherst

This work examines the influence of representation in inductive approaches to machine learning. Different representational formalisms have different biases, making some concepts easier to acquire in one formalism than in another. For instance, perceptrons (one-layer neural networks) can only learn to distinguish linearly separable concepts, whereas decision trees can make more complex types of discriminations. However, decision trees can only divide the space of instances in specific ways, which necessitates complex descriptions (and many training instances) for some concepts that are easy to represent and learn using perceptrons.

There are two main foci to Utgoff's work. The first centers on developing hybrid formalisms, which combine desirable features from two or more representational schemes. For example, his work on perceptron trees shows that one can augment each terminal node of a decision tree with a perceptron, and that in many cases this gives a simpler concept description and more rapid learning. He plans to augment this technique with the ability to incorporate yet other formalisms. The second effort, to begin in 1991, will explore approaches to representation change, in which the learner incrementally alters its descriptive language with experience.

These approaches to induction can be applied to classification and diagnostic tasks (as Quinlan and Fisher have done), but they can also be used in problem solving. To this end, Utgoff plans to employ a state-space paradigm in which one must decide which states to expand at each stage of the search process. As in Laird's work on SOAR, and Minton and Carbonell's work on PRODIGY, he plans to encode decision knowledge in terms of preference rules that give some states precedence over others. However, rather than using explanation-based methods to acquire this knowledge, he plans to learn search-control knowledge using incremental induction techniques, like those in his earlier work on decision trees and perceptron trees. This framework could be used on any task that one can view in terms of state-space search, such as scheduling problems for the Hubble space telescope or for shuttle launches.

Milestones:

1990: • Extend the perceptron tree method and evaluate its learning ability on classification tasks.
• Adapt the method to acquire preference rules for state-space search.

1991: • Evaluate methods for learning preference rules on search problems, and begin adapting it to a NASA scheduling domain.
• Initiate research on representation change.

1992: • Continue evaluation of approaches to learning preference rules, including systematic tests on a NASA domain.
• Evaluate and extend approaches to representation change.
• Begin systematic evaluation of a learning scheduler, with and without components for representation change.

1993: • Continue evaluation and improvement of learning scheduler and begin adapting the for field tests.

1994: • Application of learning scheduler to actual NASA scheduling tasks and comparison to existing fielded systems.

5. Capture, Integration, and Preservation of Life Cycle Knowledge

5.1 Large Knowledge Bases

A major portion of time and effort involved in building knowledge-based systems to support NASA's science and engineering applications goes into construction and maintenance of the knowledge base. Typically, the knowledge base is custom-crafted for the particular application task (e.g., diagnosis or design of some engineered system), and cannot be reused to perform other tasks. Rather than construct a separate knowledge base for each task, it makes sense to try and construct a large, reusable knowledge base that contains the union of the separate knowledge bases and supports several tasks at once. This would avoid duplication of knowledge engineering effort and promote uniformity and consistency in knowledge base construction.

There are two fundamental obstacles to the large knowledge base approach. First, to support multiple tasks, a knowledge base must represent far more information and go into far greater detail about the relevant engineered system. This points to a need for research on representing "first-principles" knowledge about engineered systems. Second, as the amount of detailed knowledge increases, it becomes increasingly inefficient to execute the required reasoning tasks. This points to a need for research on increasing representational and reasoning efficiency using various "knowledge compilation" techniques involving simplification, abstraction, and approximation. The following two projects are focused on these two research problems, respectively.

Multi-Use Knowledge Bases
Edward A. Feigenbaum, Yumi Iwasaki, Tom Gruber, Pandu Nayak;
Stanford University

The goal of this project is to enable the construction, maintenance, and use of knowledge bases that contain comprehensive information about the structure and function of large-scale physical devices. This entails research in the following topics: representing and reasoning with models of physical devices; using these models to simulate and explain the behavior of devices; representing and acquiring assumptions and design rationale underlying the design of devices; and, managing all of the knowledge over the life-cycle of the device. Of particular importance is the need to efficiently use the resulting device knowledge base for many purposes, ranging from knowledge retrieval by humans to automated systems for diagnosis and re-design. The project also has the goal of building a Device Modeling Environment (DME) that is a domain-independent modeling tool supporting qualitative and quantitative representations. All work to date has been done in the context of the Hubble Space Telescope (HST) and in close collaboration with the knowledge compilation task described below.

During 1990, progress was made in two major areas: knowledge representation and the DME. The electrical power system (EPS) of HST served as the domain for all of the research. In the knowledge representation area, we began to generalize prior work in the HST pointing and control system to the EPS. Work was initiated on how to maintain and reason with multiple device models, corresponding to multiple levels of structural and behavioral abstraction. We also began research on representing, modeling, and simulating the electrochemical processes that lead to a decrease in the storage capacity of the Space Telescope's batteries; this was the first attempt by the project to work with complex process knowledge. As part of the process of tool construction, we undertook a comprehensive evaluation of possible knowledge representation environments, including KEE, CYCL (the system developed by Lenat's group at MCC), and public-domain systems. We selected CYCL as our choice of an underlying knowledge representation language because of the expressiveness of the CYC representation language and its powerful inference mechanisms. Finally, a prototype user interface to a device-independent representation system was developed.

For the DME, we developed an initial architecture, using KEE as the underlying knowledge representation system. The nickel-cadmium battery model was used for a first demonstration test of the architecture. Simulation technology from Kuipers' QSIM system was integrated into a second version of DME.

In 1991, we intend to continue work in both knowledge representation and the DME. The ability to reason about "order-of-magnitude" will be added; this is of particular value for knowledge learned early in the life-cycle of devices. We will develop techniques for using goal-directed information (e.g. the query to be answered by a system for information retrieval) in order to control the process of reasoning about behavior. We will begin research on a theory of text planning for machine-generated documentation of physical devices. We intend to investigate the utility of behavioral models for the purpose of explanation. Finally, we will extend the use of the DME to a non-HST domain, most likely a micro programmable electro-mechanical system (uPEMS)--a very small robot arm constructed using semiconductor fabrication technology being developed at the Stanford Center for Design Research. The newness of the technology will force the development of multiple models based on strong assumptions about the physics of devices of this class.

Milestones:

1990: • Construct an interactive knowledge representation development environment that works with the CYCL representation language and CYC knowledge base.
- Rebuild the DME to use CYC as the underlying knowledge representation system
- Adapt a state of the art natural language generation system to the Device Modeling Environment.
- Demonstrate the total system on the HST EPS.

1991: • Incorporate orders-of-magnitude reasoning into the system.

- Develop a theory of text planning for machine-generated, interactive documentation of physical devices.
- Use the DME knowledge representations to model the conceptual designs for a micro programmable electro-mechanical system.

1992:
- Incorporate into the DME alternative techniques for modeling behavior (other than qualitative simulation, which is currently the technique employed by the system).
- Investigate use of explanations of behavior produced by simulation for the purpose of generating, acquiring, and recording design justifications.

Knowledge Compilation
Rich Keller, Catherine Baudin; RIA

The goal of this work is to investigate the potential benefits of applying knowledge compilation techniques to expert systems built by and for NASA. Potential benefits include improved run-time performance, increased robustness, reduced expert system construction and maintenance costs, and enhanced system verifiability.

Most of the expert systems developed by or for NASA can be considered first generation or "compiled" expert systems; they consist of a set of task-specific associational rules developed with the aid of domain experts. First generation systems identify complex patterns in the data and take direct action based on these patterns. These systems can be contrasted with second generation or "model-based" expert systems, which reason in a more complex fashion using detailed models of the underlying application domain during problem solving. In general, first generation systems are efficient, although brittle and limited in scope; second generation systems are much less efficient, but are more general and can perform more sophisticated and accurate reasoning in a wider variety of circumstances.

We are attempting to use knowledge compilation techniques to combine the efficiency advantages of compiled expert systems with the generality and sophistication of model-based systems. Our approach has been to study how the associational rules found in first generation systems might be "compiled" from the kind of underlying models of the application domain found in second generation systems. To produce the associational rules, a program called a "knowledge compiler" takes as input a model of the application domain, and applies a variety of techniques to reduce the computational cost of reasoning with those models. These techniques include simplification, precomputation, approximation, abstraction, and macro-formation. The compiler outputs a set of efficient associational rules that are specially crafted for solving a specified reasoning task.

There are several advantages to the compilation approach. First, when an expert system needs to be changed, it should be easier to change the system's underlying models and automatically regenerate the rules than to hand-modify

the rules themselves. Thus, this methodology streamlines the often time-consuming process of expert system maintenance. Second, the knowledge captured in the underlying models may be useful for tasks other than those originally intended when the expert system was built. Using compilation, for example, the models underlying an expert system for monitoring might be applied to produce rules for diagnosis, training, simulation, etc. Third, because the rules are mechanically compiled, the problem of verifiability is eased. Finally, when compiled rules yield an incorrect inference, it should be possible for the system to "fall back" on the underlying models to reason about where the rules came from, and why they were incorrectly applied in the current situation. Then, the system should be able to recompile new rules to handle the current situation correctly.

During 1990, we have illustrated the feasibility of this approach in the context of constructing expert systems for Hubble Space Telescope (HST). In particular, we showed how portions of both diagnosis and design expert systems for HST's Reaction Wheel Assembly component could be compiled from models of its physical structure and behavior.

In 1991, we intend to continue this work on compilation by applying our techniques to expert systems currently in operation at NASA. In particular, we wish to illustrate how portions of an existing expert system rulebase might be compiled from underlying models of the application domain. Our plan is to analyze an existing first generation system rulebase, construct a set of underlying domain models, and then build a compiler to automatically regenerate a subset of the rulebase. Thereafter, changes to the expert system would be effected by changing the underlying models and recompiling, thereby realizing some of the advantages stated above. By focusing on an operational expert system, we intend to produce a useful adjunct to existing mission software while also forcing the research to confront the complexities of operational expert systems in the NASA environment.

Milestones:

1990: • Demonstrate the compilation of diagnosis and redesign rules for the HST Reaction Wheel Assembly.
1991: • Build underlying application domain model(s).
• Construct knowledge compiler for targeted rule subset: develop and implement requisite compilation techniques.
1992: • Enlarge target rule subset and extend domain model to enable compilation of more expert system rules.
• Add compilation techniques to support compilation of extended rule subset.
• Generalize/improve knowledge compiler architecture as necessary
1993: • Develop facility to trace and debug errors in compiled rules.
• Deliver knowledge compiler to maintainers of NASA expert system.
1994: • Apply knowledge compilation techniques to a different NASA expert system to test its generality.

5.2 Knowledge Acquisition and Use of Device Models for Diagnosis and Design

This section presents methods and tools for acquiring and using engineered device models for diagnosis and design/redesign related tasks. The research issues are twofold:

(1) Modeling issues: Qualitative and quantitative representation of a device's behavior, device modelling at different levels of abstraction;
(2) Knowledge acquisition issues: how to use device models to acquire knowledge for design and diagnostic types of tasks.

Design Knowledge Acquisition and Retention

Catherine Baudin, Monte Zweben; RIA
Larry Leifer, Fred Lakin; Stanford University

This work is a partnership between RIA and the Center for Design Research (CDR) at Stanford University. The overall goal of the project is to address the problem of lost information through the lifecycle of an engineered device by providing a system that can help in recording alternative designs as well as the rationale behind design choices. The system should also assist in documenting the history of design modifications by comparing how a device meets the design requirements before and after the changes. The current system has two interacting components: the Electronic Designer's Notebook (EDN) for manipulating and organizing visual documents, and a tool for modeling the behavior of physical systems that can compare how two design alternatives meet a set of requirements. All work to date has been done in the context of conceptual design for the Space Infrared Telescope Facility (SIRTF).

During 1990, an automatic summarizer has been built in the EDN. Using this, the pages of the notebook can be automatically indexed using domain independent heuristics. The objects that are referenced by the summarizer are syntactic elements such as titles, paragraphs, and characters in bold face. In addition, a user can explicitly index a document by defining "idea tags". A set of rules has been developed that reasons about the proximity of objects to infer the possible relations among them (e.g., conditions under which a title is related to a set of paragraphs or graphics).

Also, in 1990, the architecture of the justification component has been refined. The definition language that is used to define new equations and new constraints has been extended. The old version of the system that was running a qualitative process representation of the devices has been replaced by a representation that can handle both qualitative and quantitative representations of a component's behavior by propagating intervals of values through the equations (the underlying representation is implemented using the constraint manager, described above). The fixed-mirror concept of SIRTF has been re-implemented in the new formalism and has shown significant improvement over

the first versions in terms of the accuracy of the requirement evaluation. We have begun to study how the backtracking mechanism of the constraint manager can be used to model motion and time varying constraints. In the current version of our system, the user has to select the relevant alternative solutions to compare; in the future we would like to make better use of the EDN's indices to provide assistance in selecting related documents (designs). At the moment the justification component only reports the design requirements that are violated by a given design.

In 1991, we will develop a FMEA component (Failure Mode and Effect Analysis) that can reason at the conceptual level of the design. The knowledge base framework will be extended to represent time dependent constraints. In particular, we want to refine and generalize the way moving mechanisms are defined (in the current version, the representation of a moving part is generated in the constraint system by low-level domain dependent procedures). We are also investigating the possibility of integrating a QSIM type of qualitative simulation mechanism to the constraint system. Along this line, more tests have to be performed to evaluate the adequacy of interval arithmetic for modelling devices at the conceptual stage of their design (when the values of the quantities are still imprecisely defined). The justification of design modifications also needs to be completed. Finally, we will extend the VMACS query language which underlies the EDN so that documents can be retrieved on the basis of the type of components and design requirements that are referred to by the pages. This would allow a user to ask, "Give me the documents that refer to 'beam splitting device' and where the stray-light requirement is violated."

Milestones:

1990: • Demonstrate a version of the system integrating the new indexing capabilities and a refined justification component.
- Run the system on a new device that include moving parts.
- Demonstrate how the system can be used to help a designer make decisions.

1991: • Complete the knowledge based framework; use the constraint system to model dynamic systems.
- Extend and demonstrate the justification module of the system.
- Complete the indexing mechanism of VMACS to be able to analyze any visual output information that has been generated by an external program.
- Demonstrate system utility in a second engineering domain.

1992: • Extend the justification component of the system to compare alternative designs that differ in basic structure.
- Perform a deeper study of how the hypertext links of the EDN can be used to structure the information related to a particular design.
- Link the system to other external programs that perform design related tasks such as numeric simulation.

Corporate Memory Facility

Guy Boy; RIA
John Boose, Jeff Bradshaw; Boeing Advanced Technology Center

The Corporate Memory Facility (CMF) project focuses on the development of several advanced tools that capture and reason about the decision history, rationale, expertise, and experience embedded within major projects such as Space Station Freedom. A CMF can be used during the early requirements studies, during the design and manufacturing of a system, and finally during the operations of that system. Users may range from agencies developing requests for proposals, bidders responding to proposals, implementors of proposals, as well as users of the developed systems. It is intended to be a store of information useful throughout the lifecycle of a system as well as a set of tools for accessing this vast amount of information.

The CMF project began in October 1988 at the Boeing Computer Services Advanced Technology Center (ATC). In 1989, a tool that facilitated the generation, capture, and comparison of design alternatives for Space Station Freedom systems was demonstrated in the domain context of Electrical Power (EPS) and life support (ECLSS) subsystems of SSF. The AQUINAS knowledge acquisition tool was enhanced to capture designer voice trails that elucidate the rationale behind design alternatives. In 1990 there will be a demonstration of a new graphical tool that assists designers in developing design alternatives that meet design requirements. This capability will be linked to the trade-study capabilities of the first demonstration. The knowledge acquisition tool underlying the first demonstration capability was used to perform a trade study of alternatives for a power subsystem interface between automatic circuit breakers and a computer. The design engineer felt that this capability provided valuable assistance in conducting the tradeoff. Additionally, it provided a multiple designer trade off capability for ECLSS.

In 1991, we will focus on extending current capabilities in support of the generation and evaluation of design alternatives using the design of a logistics module docking port as a test domain. These results will be used to formalize a process for decision rationale capture. We will also begin the implementation of the Design Alternative Rationale Tool (DART) in the Technical and Management Information System (TMIS) of SSF. This constitutes a port of the CMF capability discussed above to the TMIS-compatible workstations.

Note that this work is co-funded by both the SSF Advanced Development Program and the SSF Level II TMIS Design Knowledge Capture effort. It represents an interesting blend of activities from basic research on knowledge acquisition to tool development to actual integration with TMIS.

Milestones:

1990: • Enhance design alternative generation and evaluation capability.
- Deliver initial version of the DART tool.
- Deliver DART documentation and video tape.

1991: • Extend DART to handle voice entry, storage, indexing and retrieval.
• Enhance DART scoring facility for alternative designs.
1992: • Automatically generate CLIPS knowledge bases from DART-captured knowledge for support of FDIR expert systems.

AI and Multi-Faceted Modeling
Bernard Zeigler, Francois Cellier, Jerzy Rozenblit; University of Arizona

The purpose of this project is to develop the concepts and tools for multi-abstraction models that can be used for a variety of purposes including design, diagnosis, and operations. Work is being done in the context of potential robot-operated laboratory experiments on Space Station Freedom. The models being developed for this project are expressed in discrete event and continuous simulation formalisms. In particular, the discrete event formalism used by Zeigler and co-workers is called DEVS (Discrete Event System Specification). It is based on set theory and was developed in the early 1970s. The same organizational principles will be extended to include abstractions that can be expressed in qualitative reasoning and planning formalisms.

Relative to qualitative formalisms, DEVS device modeling has superior design and diagnosis features where the system is well understood from the standpoint of traditional differential/difference equations. Relative to planning, simulation can provide detailed, realistic, models for better evaluation of plans based on less detailed abstractions.

In 1990, we have built models for several classes of entities in the laboratory robot domain. These include the robots themselves, scientific instruments and other devices, and a world map which shows the locations and orientations of objects within the laboratory space. We have developed both static and dynamic models in this environment. Static, or memoryless, models represent relations in a modelled system that are invariant over time. For example, image generators for objects answer questions concerning how objects look from various points of view. Dynamic models are those whose actions is history dependent. Such models can be given a state representation in which all history is captured in a single (perhaps complex) state.

Our approach differs from previous work in physical system modeling in the following respects:

1. The abstractions employed for design, planning and diagnosis are derived in a systematic way from a base model which is a detailed, presumably more realistic, model of the system. The approach produces, as a product, morphism relations linking the various models so that when one model must be modified, the others are modifiable in a consistent manner.

2. Discrete event and continuous modelling formalisms are used, in particular DEVS models, which can simulate any computable model.

3. The capability of AI techniques to generate plans are combined with the ability of simulation to evaluate them.

Current work focuses on developing a design model for the Robot Architecture, i.e., the cognitive control, internal models, sensory and movement capabilities needed to execute fluid handling experiments. A suite of Model Plan Units (MPUs) has been developed with the capability to execute some basic lab procedures such as filling bottles with liquids, bringing them to a work site, mixing them together, bringing the mixture to a heater and storing the result. Plans for directing such procedures to execute complete experiments and for fault recovery are embedded in a task-orderer MPU. This is an MPU with behavior much the same as other MPUs except that its model concerns the state of an experiment as a whole rather than that of any device.

In 1991, we intend to concentrate on bringing all of the tools we have developed together into a Coherent Multi-Abstraction Model Base that incorporates autonomous system design and maintenance. At a minimum, this system will include the DEVS work discussed above along with context-sensitive model pruning and abstraction development tools that leave a usable record of how the abstractions were developed.

Milestones:

1990: • Complete first stage of the Simulation Environment for Robot-Managed Laboratory.
• Demonstrate how design knowledge can be used in planning, operations, and diagnosis.

1991: • Complete the prototype of the Coherent Multi-Abstraction Model Base.
• Demonstrate how the multi-abstraction models can be used to support design.

1992: • Complete the second stage of the Simulation Environment for Robot-managed Laboratory, including the incorporation of the Coherent Multi-Abstraction Model Base within the robot architecture design.
• Extend the system to include the model of human scientist at the upper layer, fault recovery and repair capability, and enhanced fluid handling.

Learning in Diagnosis

Deepak Kulkarni; RIA

This project conducts research in learning knowledge useful for the diagnosis of a device using a concept formation system. Observations of the behavior of a device consist of the readings of sensors at a certain sampling rate. Diagnosing the fault directly from the raw data is computationally intractable. To make the complexity of the reasoning manageable, the behavior of the device is is described as a sequence of qualitative states. In the cases where real data is

not available, a qualitative simulator is used to produce instances of device behavior under the normal and faulty conditions.

This work uses LABYRINTH , a concept formation system for structured objects, to learn diagnostic knowledge from the instances of device behavior, and to use the system to diagnose a fault. LABYRINTH learns concepts for faults, and for classes of faults. It tries to match the observed dynamic behavior of a device against these concepts, and retrieves a number of fault hypotheses. A fault hypothesis can be that H1, a specific heater, is not working. Alternatively, it can refer to a class of faults, e.g., one of the heaters in a component is not working. We believe that an advantage of using LABYRINTH is its accuracy in making such specific or general diagnoses accurately from the observations. As the reading of the sensors are taken after a fixed interval of time, the observations may miss some changes in the behavior of the device. Furthermore there may be inaccuracies in the readings of the sensors. LABYRINTH is capable of dealing with observations that miss some changes in the behavior, and that have considerable noise.

Work in 1990 includes the use of active experimentation to prune candidate hypotheses. An experiment is a sequence of actions that, when executed, will allow discrimination between hypotheses. The system searches for an experiment for which the predicted behavior of the device is different for the alternative hypotheses. When the experiment is executed, the observed behavior of the device is used to prune out suspect hypotheses. The novelty of this work is that it uses qualitative states, a concept formation system for structured objects, and active experimentation in the diagnosis from observations of dynamic behavior of a device.

In 1991 we will extend the research work to allow for the discrimination among competing hypotheses based upon automatically learned diagnostic knowledge. We will begin to apply the work to a real NASA domain; our current plan is to use the Space Station Freedom Thermal Control Expert System and the extensive library of potential failures studies developed by that project.

Milestones:

1990: • Develop mechanism for translating quantitative observations into qualitative states implemented for a simple heating device.
• Develop the concept formation component completed.
• Demonstrate the system to retrieve multiple fault hypotheses.
1991: • Extend system to plan experiments to discriminate between hypotheses.
• Extend system to use observed results of experiments to prune hypotheses.
• Initiate application to the SADP Thermal Control System.
• Refine the mechanism for translating quantitative states into qualitative states for the TCS data.
1992: • Extend system to use a qualitative simulator to predict behaviors.

- Refine the concept formation system for diagnosis of TCS.
- Demonstrate the system's ability to retrieve multiple hypotheses.
- Demonstrate the experimentation component for use on TCS.

1993: • Apply the work to an on-flight subsystem.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	Dates attached	

4. TITLE AND SUBTITLE

Titles/Authors - Attached

5. FUNDING NUMBERS

6. AUTHOR(S)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Code FIA - Artificial Intelligence Research Branch
Information Sciences Division

8. PERFORMING ORGANIZATION REPORT NUMBER

Attached

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Nasa/Ames Research Center

Moffett Field, CA. 94035-1000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Available for Public Distribution

Peter Friedland 5/14/92 BRANCH CHIEF

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Abstracts ATTACHED

14. SUBJECT TERMS

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT