PROPOSAL FOR CONTINUATION OF THE STANFORD ARTIFICIAL INTELLIGENCE PROJECT

JOHN MCCARTHY, Professor of Computer Science
Principal Investigator

ARTHUR SAMUEL, Senior Research Associate in Computer Science
Associate Investigator

and

THE HEURISTIC DENDRAL PROJECT

EDWARD FEIGENBAUM, Professor of Computer Science
Co-Principal Investigator

JOSHUA LEDERBERG, Professor of Genetics
Co-Principal Investigator

ARPA SD-183
JUNE 1969

PROPOSAL FOR CONTINUATION OF THE STANFORD ARTIFICIAL INTELLIGENCE PROJECT

JOHN McCARTHY, Professor of Computer Science
Principal Investigator

ARTHUR SAMUEL, Senior Research Associate in Computer Science
Associate Investigator

and

THE HEURISTIC DENDRAL PROJECT

EDWARD FEIGENBAUM, Professor of Computer Science
Co-Principal Investigator

JOSHUA LEDERBERG, Professor of Genetics
Co-Principal Investigator

ARPA SD-183

June, 1969

ABSTRACT

$1,975,859 is requested to continue research in artificial intelligence and related theoretical work in computer science for an eighteen month period beginning 1 January 1970. Principal research objectives are in the areas of representation theory, mathematical theory of computation, machine interaction with the physical world, computer recognition of speech, heuristic search strategies, and models of cognitive processes.

Table of Contents

1. Introduction

The work of the Stanford Artificial Intelligence Project is charted in Figure 1.

In this introduction we shall make a few remarks about the outline as a whole, then comment on the status of the subdivisions and our plans for future work in them. More detailed descriptions and plans will be given later in the proposal.

This is not the place for a full discussion of the present state of research in artificial intelligence. Instead, we shall make a few statements about the general situation into which this proposal fits.

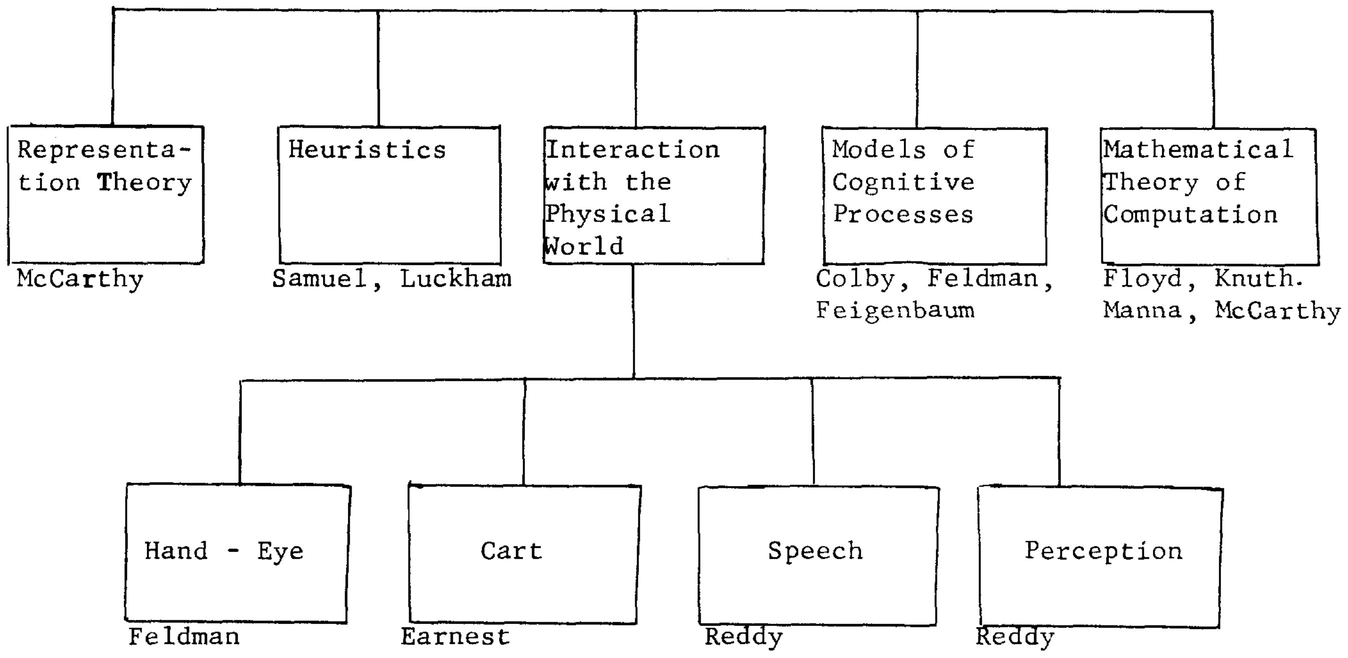
1. Artificial intelligence is the experimental and theoretical study of perceptual and intellectual processes using computers. Its ultimate goal is to understand these processes well enough to make a computer perceive, understand and act in ways now only possible for humans.

2. The information for this study comes in part from observation of human behavior, including self-observation, but mainly from experiments with programs designed to solve problems chosen to require the intellectual processes under study.

3. This understanding is at present in a very preliminary state, no one can say how long it will take to reach the understanding required to duplicate human intellectual performance because there are fundamental discoveries yet to be made.

4. Nevertheless, progress in identifying and duplicating intellectual mechanisms is being made and the range of problems that computers can be made to solve is increasing.

Figure 1. Structure of the Stanford Artificial Intelligence Project


Representation Theory
McCarthy
Heuristics
Samuel, Luckham
Interaction with the Physical World
Models of Cognitive Processes
Colby, Feldman, Feigenbaum
Mathematical Theory of Computation
Floyd, Knuth. Manna, McCarthy
Hand - Eye
Feldman
Cart
Earnest
Speech
Reddy
Perception
Reddy

5. Many blind alleys have been and are being followed and many mistakes are being made. Nevertheless, a body of fundamental knowledge is accumulating.

6. An important limitation up to the present has been the lack of well-trained and well-motivated scientists, working on artificial intelligence. The graduate program in computer science at Stanford has begun to change this situation.

7. Although full solution of the artificial intelligence problem is unpredictably far off, the understanding so far achieved has important potential practical applications. The development of these applications is worth undertaking.

We have divided our work in artificial intelligence into four categories: epistemology (representation theory), heuristics, interaction with the physical world, and models of cognitive processes.

The identification of the epistemological and heuristic parts of the artificial intelligence problems as separate entities is a result of work of the last few years, mainly by John McCarthy. The epistemological part is to choose a suitable representation for situations and the rules that describe how situations change. This description must be general enough to cover all problem solving situations, and even more important, it must be able to express all likely states of knowledge of the situation and the rules by which it changes spontaneously or by the actions of the problem solver.

The heuristic part of the artificial intelligence problem is to devise methods that, starting from a suitable representation of the information, will express explicitly the actions that have to be performed e.g., find the right chess move or the right next step in a proof. Work in heuristics has been carried on in many places for a number of years. Here, Samuel is leading work applying learning methods and Luckham is leading work in theorem proving by computer.

Samuel's new version of the checker program with learning is working and methods from that program are now being applied to learning in speech recognition. Luckham has been able to include in his resolution program almost all the methods found useful by others together with new methods of his own. Details on the results and future plans of the work are given in Section 5, below.

Much of our work involves modeling cognitive processes in one way or another, but a concentrated attack on this is through Feigenbaum's and Lederberg's Heuristic Dendral Project. This involves getting chemists to express their rules of reasonableness for the structure of organic compounds in a way that can be used by the computer program. Colby's work on belief systems and Feldman's language research are also aimed at developing cognitive models.

The largest area of work of the Stanford Artificial Intelligence Project both in terms of people and money, has been computer interaction with the physical world.

This work has gone slower than was anticipated when the project started. There are two reasons for this. First, it was more difficult, expensive, and time-consuming than expected to get a time-shared facility capable of both real-time and ordinary time-sharing. This task is substantially complete; all but reliability problems appear to be solved.

The second problem is that it has taken a long time to develop a group of good computer scientists whose main interest is the vision problem. The situation has greatly improved since the major hardware problems have been solved and since Professor Jerome Feldman has made hand-eye his major area of concentration. Now ten graduate students (Ruzena Bajcsyova, Gil Falk, Gunnar Grape, Lou Paul, Irwin Sobel, Jay Tennenbaum, Dave VanVoorhis, Bruce Baumgart, Rod Schmidt and Jack Buchanan) are doing research in this and other perception areas that should lead to theses.

Finally, in our work on the mathematical theory of computation, substantial results have been obtained by McCarthy and his students and more recently by Manna and Pnueli. The Stanford work in this area has been further strengthened by the addition of Robert Floyd and Donald Knuth to the faculty.

The balance of this proposal is divided into sections that discuss plans in each research area. The current research staff is given here for each topic.

<u>Theory</u>

<u>Representation Theory</u>

Fred Goldstein	Prof. John McCarthy

<u>Mathematical Theory of Computation</u>

Prof. Robert Floyd	Steven Ness
Prof. Donald Knuth	Prof. Zohar Manna
Lockwood Morris	Prof. John McCarthy

Visual Perception and Control

Hand-Eye Systems

Gilbert Falk
Prof. Jerome Feldman
Gunnar Grape
Richard Paul
Karl Pingle
Gerald Shapiro
Irwin Sobel
Jay Tenenbaum

Visual Perception

Ruzena Bajcsyova
Dr. Manfred Hueckel
Michael Kelly
Prof. D. Raj Reddy
David VanVoorhis

Visual Control of a Vehicle

Bruce Baumgart
Jack Buchanan
Lester Earnest
Rodney Schmidt

Speech Recognition

Dr. James Beauchamp
Lee Erman
Gary Goodman
Richard Neely
Stephen Plant
Prof. D. Raj Reddy

Heuristic Search

Machine Learning

Dr. Morton Astrahan
Johathan Ryder
Dr. Arthur Samuel
Joseph Siberz
Dr. George White

Theorem Proving

John Allen
Dr. David Luckham
Dr. Nils Nilsson

Models of Cognitive Processes

Heuristic Dendral

Prof. Malcolm Bersohn	Dr. Allan Duffield
Dennis Brown	Isu Fang
Dr. Bruce Buchanan	Prof. Edward Feigenbaum
Allan Delfino	Dr. Gustav Schroll
Prof. Carl Djerassi	Georgia Sutherland

Language Research

Prof. Jerome Feldman	Stephen Reder

Higher Mental Functions

Dr. Kenneth Colby	David Smith
Dr. Franklin Hilf	Lawrence Tesler
Dr. Roger Schank	Sylvia Weber

2. Theory

2.1 Representation Theory

Our recent theoretical work on artificial intelligence has led us to divide the subject into two parts: representation theory and heuristics. This division arises as follows.

When we try to make a computer program that solves a certain class of problems, our first task is to decide what information is involved in stating the problem and is available to help in its solution. Next we must decide how this information is to be represented in the memory of the computer. Only then can we choose the algorithms for manipulating this information to solve our problem. Representation theory deals with what information we need and how it is represented in the computer. Heuristics is concerned with the structure of the problem solving algorithms.

In the past, work in artificial intelligence has been content with a rather perfunctory approach to representations. A representation is chosen rather quickly for a class of problems and then all attention is turned to devising, programming and testing heuristics. The trouble with this approach is that the resulting programs lack generality and are not readily modifiable to attack new classes of problems.

The first goal of representation theory is to devise a general way of representing information in the computer. It should be capable of representing any state of partial information that a person might have about a problem and the general information necessary to solve it. In (1958) McCarthy posed the problem of getting a program with common sense in approximately these terms and suggested using sentences in an appropriate formal language to represent what the program knows. The advantage of representing information by sentences is that sentences have other sentences as logical consequences and the program can find consequences relevant to the goals at hand. Thus, representation of information by sentences allows the following:

1. A person can instruct the system without detailed knowledge of what sentences are already in the memory. Namely, the procedures for solving a problem using information in sentence form do not require that the information be in a particular order nor even a particular grouping of the information into sentences. All they require is that what to do is a logical consequence of the collection of sentences.

2. Similar considerations apply to information generated by the program itself.

3. Representing information by sentences seems to be the only clean way of separating that information which is common knowledge and so should be already in the system from information about a particular problem.

On the other hand, because each sentence has to carry with it much of its frame of reference, representation of information by sentences is very voluminous. It seems clear that other forms of information (e.g., tables) must also be used but the content of these other forms should be described by sentences.

In the last ten years considerable progress has been made in the use of the sentence representation. In the heuristic direction, theorem proving and problem solving programs based on J. Allen Robinson's resolution have been designed and continously improved. Cordell Green's QA3 (developed at the Stanford Research Institute and described in Reference 2,) and David Luckham's (this project) represent the present state of the art. Secondly, the theory of how to represent facts concerning causality, ability and knowledge for artificial intelligence purposes has been developed, mainly by McCarthy and his students. (McCarthy and Hayes, 1969) gives a good summary of where this work stands. The connection between this work and the subject of philosophical logic has been established, and some of the more recent efforts of philosophers to understand concepts of causality, ability and knowledge turn out to be of use in trying to make a computer use them. Another connection has been established with work in mathematical theory of computation. Namely, the results of McCarthy, Floyd and Manna about the correctness of computer programs can be used to make a system that finds strategies for achieving goals.

Our research work in representation by sentences is continuing along the following lines.

1. Improve the formal treatments in (McCarthy and Hayes, 1969) of the concepts of causality, ability and knowledge so as to get more realistic expressions of the "common knowledge" of these concepts.

2. Work on more realistic examples.

3. Translate the modal logic of the present formalism into first order logic so that present theorem proving and problem solving programs can be used.

4. Connect the work on representations with our work on representation of visual information where representation by sentences is rather clearly inappropriate.

Problems of representation also arise in our work with vision. In order to assemble an object out of parts or to drive a vehicle, the computer must have a suitable representation of the scene in its memory.

In the assembly or hand-eye case the representation of choice is a list of objects in the environment giving for each its shape (e.g., for objects with flat faces as a list of faces described by edges and vertices in turn) and its location. In the driving case we face another problem. Namely, most of the scene has to be dismissed as irrelevant. Thus a tree must be recognized and dismissed without generating a list of branches, twigs and leaves. That there are no obstacles on the road has to be verified with high reliability. Therefore, the representation in memory of a road scene must start with a division of the scan into "blobs" most of which are coarsely identified and dismissed while others are described in greater detail.

A project for such a description is being worked out along the following lines: The scene is divided into regions r in such a way as to minimize a quantity c that we call the complexity of the description. Letting R be the set of regions, we write

$$c = \sum_{r \in R} [a \times \text{perimeter}(r) + b \times \text{area}(r) \times \text{inhomogeneity}(r) + 1]$$

In the simplest case the inhomogeneity might be the variance of the brightness of the points in the region or the variance over the region of the vector of light intensities seen through three color filters. If the parameter b is small the optimal description will tend to have few regions so as to minimize the lengths of the boundaries. Thus a tree will appear as a single blob and the inhomogeneity resulting from the blue of sky, green of leaf, and brown of branches and trunk will be suffered. If the coefficient b is increased then the optimal description will separate out the trunks and use a longer boundary to separate the tree from the sky. A very large value of b would be required to make the system separate out each leaf and twig.

References

1. John McCarthy, "The Advice Taker" in Mechanisation of Thought Processes, Vol. 1, pp 77-84, Proc. Symposium, National Physical Laboratory, London, 1958. Reprinted in M. Minsky (ed), Semantic Information Processing, MIT Press, Cambridge, 1968.

2. Cordell Green, "The application of Theorem Proving to Question Answering Systems", Ph.D. Thesis in Electrical Engineering, Stanford University, 1969.

3. John McCarthy and Patrick Hayes, "Some Philosophical Problems from the Standpoint of Artificial Intelligence" in D. Michie (ed), Machine Intelligence 4, American Elsevier, New York, 1969.

2.2 Mathematical Theory of Computation

What are the fundamental properties and relations of algorithms, the data structures on which they operate, the programming languages in which they are expressed, the problems they are written to solve, and the computers that execute them? In our view, some of these fundamental properties and relations are:

1. The relations between the input and output information of an algorithm.

2. Whether the algorithm ever terminates for inputs satisfying certain conditions.

3. The equivalence of two algorithms.

4. Transformations of algorithms that preserve equivalence (and perhaps increase speed).

5. Relations between algorithms and the speed with which computers can carry them out.

6. The semantics of programming languages: i.e., the relation between the form of a program and properties of the algorithm it carries out.

The goal of our research has been a formal theory in which the relevant properties of actual algorithms can be stated and proved and the proofs checked by computer. Attainment of this goal would allow the replacement of debugging programs, (a never-ending process) by debugging proofs that programs have desired properties (a process that terminates conclusively when the proof-checker program accepts a proof that the given program has the desired properties).

These goals were first stated in (McCarthy, 1962) and progress was registered in that paper and in others by McCarthy, Painter, and

Kaplan (References 1-7). Results on correctness of programs and convergence in a different formalism were obtained by Floyd (1967), and Manna (1968). We should also mention the work of the IBM Vienna group in extending the methods and ideas of (McCarthy 1963, 1965) and applying them to PL/1. Now both Robert Floyd and Zohar Manna have joined the Stanford Computer Science Department and the Artificial Intelligence Project.

Within the last year Manna, McCarthy and Amir Pnueli have united the formalism of McCarthy with that of Floyd and have applied extensions of Manna's previous results to several concrete algorithms.

At present the part of mathematical theory of computation based on first order logic has reached a certain stage of completeness and Manna and McCarthy plan to write a book about it. Extensions of the formalism using set theory are being developed and a beginning has been made in extending Manna's method to parallel processes.

The first connections of this theory to the theorem proving research in artificial intelligence have been made by Green at Stanford and SRI. Green has shown how his theorem prover QA3 can be made to find a LISP program for sorting a list.

REFERENCES

1. John McCarthy, "A Basis for a Mathematical Theory of Computation", in P. Biaffort and D. Hershberg (eds), Computer Programming and Formal Systems, North-Holland, Amsterdam, 1963.

2. John McCarthy, "Towards a Mathematical Theory of Computation" in Proc. IFIP Congress 62, North-Holland, Amsterdam, 1963.

3. John McCarthy, "A Formal Description of a Subset of Algol" in T. Steele (ed), Formal Language Description Languages, North-Holland, Amsterdam, 1966.

4. John McCarthy, "Problems in the Theory of Computation", in Proc. IFIP Congress 65, Spartan, Washington D.C., 1965.

5. James Painter, "Semantic Correctness of a Compiler for an Algol-like Language", Ph.D. Thesis in Computer Science, Stanford University, 1967, (Memo AI-44).

6. Don Kaplan, "Regular Expressions and the Equivalence of Programs", Ph.D. Thesis in Computer Science (Memo AI-63), Stanford University, July 1968.

7. Don Kaplan, "Some Completeness Results in the Mathematical Theory of Computation", J. ACM, January, 1968.

8. Robert Floyd, "Assigning Meanings to Programs", in Proc. Symposia in Applied Math., Vol. XIX, American Mathematical Society, Providence, 1967.

9. Zohar Manna, "Termination of Algorithms", Ph.D. Thesis, Computer Science Department, Carnegie-Mellon University, Pittsburg, 1968.

10. Zohar Manna, "Properties of Programs and the First Order Predicate Calculus", J. ACM, April 1969.

11. Zohar Manna, "Formalization of Properties of Programs", J. System and Computer Sciences, May 1969.

12. Zohar Manna and Amir Pnueli, "Formalization of Properties of Recursively Defined Functions", Proc. ACM Symposium on Computing Theory, May 1969.

3. Visual Perception and Control

The overall goal of the hand-eye project is to design and implement a system which exhibits intelligent perceptual-motor behavior. An important subgoal is that the problems that arise in the design of system components be solved in ways which are sufficiently general to be scientifically interesting. Thus, for example, we have put considerable effort into understanding depth perception although the special environment we are using may allow for ad hoc solutions.

More specifically, our goals for the coming year include:

1) Theories and programs to solve basic recognition subtasks such as edge following, corner finding, object identification, etc.

2) A system which allows basic routines to be assembled into embodiments of perceptual and manipulative strategies.

3) A means of studying and using the interactions of the various basic processes.

To provide a sense of direction and to bound our aspirations, we proposed a class of tasks which we hope to have the hand-eye performing by 1971.

Two preliminary tasks are:

1) The ability to move the mechanical arm precisely by making use of visual feedback, and

2) The ability to pick up a specified object in a complicated scene and orient it.

These two tasks are prerequisites for our main task - the building of fairly complex constructions (castles) out of simple blocks. The blocks are restricted to being plane-bounded and convex. The castle might be explicitly described by a set of associations relating its sub-parts or we might simply be given one or more views of it.

One of the most interesting aspects of this task is the various levels of feedback which can be used in the building process. In some cases, one need only know that a block is still in place, whereupon tactile feedback is sufficient. If the situation is more critical one might visually determine the placement error and alter the remainder of the strategy accordingly. Finally, there is the possibility of adjusting the block, under visual control, until the error is sufficiently small [23].

The use of visual feedback in block stacking presents a rather different problem than those normally discussed in picture processing. The vision routine has the job of determining the accuracy with which some block was placed. The total scene may be very complicated and it would be absurd to perform a complete scene analysis. Furthermore, the properties of the blocks to be examined may be known in great detail and the vision routine would be able to take advantage of this fact. This example typifies the core problem: context - sensitive visual perception.

3.1 The Organization of a Visual Perception System

Perception, and most particularly visual perception, is a complex process requiring a system which is sensitive to all the various levels of detail of the environment. Furthermore, since the available data is potentially over whelming the system must have both the mechanisms and appropriate strategies to select what data are worthy of its attention and what level of detail is best suited to the current perceptual goal.

Our approach to the system design centers on two basic issues:

1) Levels of detail, and

2) strategies for attention.

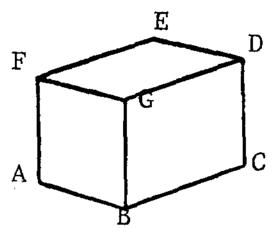
Data from a scene may be structured to varying degrees. At the lowest level lie the intensity and color of the light at a particular point in the visual field at a higher level are those objects in the visual scene which we dignify by the use of nouns; at a still higher level one notices interrelationships and relative motion between objects. At the highest level one is aware of the total situation -- as "Danger. Collision imminent." Ordinarily, we are conscious only of our perceptions of objects and situations, but the fact that we can learn to draw indicates that lower level details are perceived and can be made accessible to consciousness. It is curious that we must <u>learn</u> to draw -- as if the lower levels of visual patterns are coalesced into objects at a preconscious level. This notion gives rise to a simplified theory of perception held by many workers in perception and pattern recognition. The theory is embodied in a strategy of perception which places attention first at the lowest level of detail and then extracts successively higher levels until the organization of the entire scene is understood. Thus by processing intensity and color distributions one obtains texture, edges, and corners. From this information, regions are extracted and these in turn are associated into bodies. Then the bodies are identified as objects and their various interrelationships are derived. Thus:

Level: ^{1}points $\rightarrow$ ^{2}lines $\rightarrow$ ^{3}regions $\rightarrow$ ^{4}bodies $\rightarrow$ ^{5}objects $\rightarrow$ ^{6}scene

Essentially, all the early work on visual perception, including our own, proceeded along these lines. To some extent, the work of Guzman [11] on finding the distinct bodies in a perfect line drawing (level $2 \twoheadrightarrow$ level 4) had an undesirable effect on the field. Guzman's program was so successful that it sent people on a quest for the perfect line drawing program. Although we have had considerable success [7, 14] at generating line-drawings, it has become apparent that the strict bottom-to-top processing sequence is not optimal.

As an alternative to this linear approach, the model of vision which we find useful cooperativly involves analysis at various levels in an attempt to understand a scene. There is a large body of psychological evidence [4] indicating the dependence of perception upon global information and upon preconceived ideas. Although many of the well known optical illusions fall in this class, one can also show that there are simple scenes which are ambiguous in the absence of global inoformation, but are easily resolved in context.

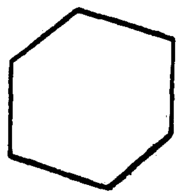
A most striking case of this is the ground plane assumption [15], which has become a cornerstone of all robot perceptual systems. From a monocular image it is impossible, in general, to calculate the distance of an object from the camera. If, however, the object is lying on a known plane (one whose transformation to image coordinates is available) then the depth of the object's base vertices is known. This particular piece of global information has been implicitly used for depth information, but has many other uses. Consider the following line drawing:



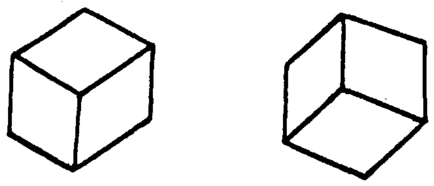
If one knew that this object were lying on the plane determined by ABC which is known, then one would know the projection of each point in the image onto the ABC plane. Each point, e.g., F must be on the line determined by its projection onto the ABC plane and the lens center. If the line AF is perpendicular to the plane we then know the length of AF.

Further, we can often determine whether or not AF is perpendicular to the plane from the information available. The lens center, point A and the projection of point F determine a plane, which contains the line AF. If this plane is perpendicular to ABC then the line AF is also, for objects which are at all regular [15]. If one knew the lengths of AF, BG, and CD and their angles with the ABC plane, then the coordinates of F, G, and D are computable and assuming F, G, D and E are in a plane is sufficient to determine E. More technically, the assumptions we have made allow only one degree of freedom in the choice of plane-bounded convex objects which could yield this image. Thus, the ground plane hypothesis plus some global regularity conditions allow for the complete description of an object from a single monocular view. Of course, these conditions may not hold, but we have some encouraging preliminary results in object recognition using these kinds of techniques.

A somewhat more basic problem arises in the consideration of the following image:



which might have come from, among other things:



The interior edges might very well be less distinct and be missed by the program which first tried to form a line drawing. At some higher perceptual level, a program could detect the ambiguity and attempt to find the interior edges. With the contextual information available, the system could then use highly specialized tests to determine the presence of an edge. Further, since the area involved is relatively small, it might also be reasonable to apply very sensitive general operations which are too costly to use on an entire scene.

In both examples we see how an organization which utilizes selective attention may facilitate perception. A vision system which worked strictly bottom-to-top would have no notion of attention. There would be a standard line finding operation, followed by an attempt to fit intersections, etc. <u>In that kind of system there are inherent limitations [17] in balancing noise sensitivity against ability to perceive detail</u>.

In addition to the enhanced perceptual ability attained by selective attention there are cost-benefit advantages. The vision programs will contain a wide variety of routines for analyzing various properties of pictures - any system which attempted to use all of them on each picture would be hopelessly inefficient.

The goal, then is to produce a flexible visual perception system capable of selective attention and of integrating information from all levels of perception. An obvious prerequisite for such a system is a monitor, language, and data structure capable of its support. Our proposed design is described in Section 3.2.

A second necessary ingredient of any cooperative system is a large set of flexible basic vision routines. Among the necessary functions are: reading raw data, changing the camera position and parameters, edge finding, corner fitting, region finding, analysis into distinct bodies, identification of particular objects, and complete scene analysis.

3.2 Hand-Eye Systems

Our first hand-eye system used many ad hoc solutions and was mainly concerned with the problems of combining the minimum necessary hardware and software components. This primitive, but complete system for block-stacking under visual control was completed in May, 1967 and has been described elsewhere [14]. The functional diagram of Figure 2 provides a sufficient description for our purposes. Our most recent work has involved the redesign of the system configuration and more careful study of each of the component programs.

The system configuration must be able to support a large number of program modules communicating with the world and with each other in complex ways. To achieve this capability we are undertaking a rather ambitious system-programming project including a submonitor, a high-level language, and a new data structure. The goal of this project is to produce a hand-eye laboratory in which it will be relativly easy to experiment with new ideas in perception, modeling, problem-solving and control. This laboratory will also, hopefully, provide a testing ground for many related artificial intelligence problems and should be relevant to the design of any large flexible real-time system.

The hand-eye laboratory will have to accomodate programs whose total size is several times the size of core memory. Further, as we have shown in Section 3.1, the order in which these programs are executed cannot be determined in advance. These programs must be able to communicate with each other and with a common global model which represents the system's knowledge of the world. Since many operations require moving physical devices (like the arm and camera) which entail long delays, we would like to allow parallel execution of hand-eye sub-programs. We are implementing

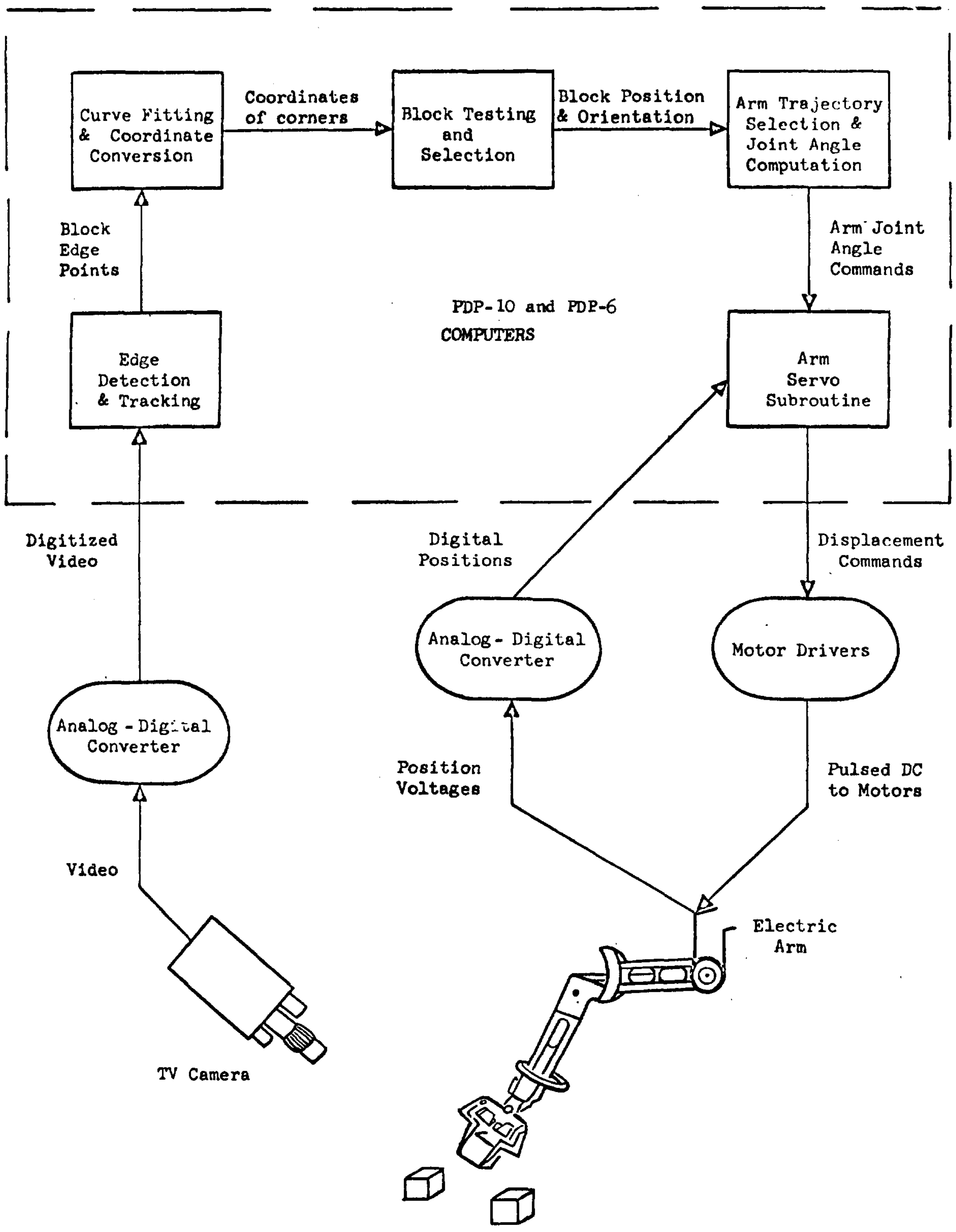


Figure 2 The Initial Block-Stacking System

a submonitor to perform these functions, handle messages, and carry out changes to the global model.

The language and data-structure designs are closely tied to the sub-monitor and to each other. The language will be an extension of our ALGOL Compiler [21] along the lines of the associative language, LEAP [3]. The central concept of LEAP and the underlying data structure is the association: attribute·object = value. The use of associations for world-modeling is described in detail in [9]. An important new concept of this version of LEAP is the use of local and global associative structures. The global structure contains the world model shared jointly among all programs working to analyze a scene. Every atomic object (item) is either local or global; the associative structure local to a subprogram may contain associations including global items, but not vice-versa. Any attempt to alter the global associative structure is trapped to the sub-monitor which determines when the alteration should be allowed. The language will contain primitives for local and global associations, message handling and interrupt processing.

3.3 Visual Perception Projects

One important aspect of the general vision system is the adaptation of the input mechanisms to the visual environment. Selective attention must be implemented at the level of the vision hardware by choosing accomodative strategies which reflect current perceptual goals. For example, the camera could be sensitized to a specific color characteristic of a desired object (via a color filter). This may yield a gross reduction in the volume of information which must be processed.

The camera parameters currently under computer control are the pan and tilt angles, lens choice (turret) and gray scale range. There are two hard problems in adaptation which arise from the need for a common world model. When the camera is panned, it gets a new view. The images of objects in this new view must be placed in correspondence with the old images of the same objects. An even more difficult problem is to compute accurately the perspective transformation [16] applicable in the new situation. Sobel [20] is developing techniques for these problems, relying heavily on the literature of photogrammetry.

A major area of interest has been the development of edge and line finders. There have been extensive analytical and practical studies of various spatial filtering and edge finding techniques [7, 22]. More recently, we have begun to look at feature verifiers which will use global information and a prediction to help identify a feature.

There are also programs which do fairly well at corner finding, region extraction, etc. These are flexible and might be incorporated into a vision system organized as we have suggested. The real problem is to develop routines for these tasks which are sensitive to possible errors and ambiguities and know when to ask for help. A related issue is the language for communicating between vision programs at various levels.

Much of the work, to date, on visual perception by machines has centered around plane-bounded man-made objects. Unfortunately, most naturally occurring objects do not have well defined edges. This raises the question of whether many of the presently used edge and object recognition techniques are likely to be useful and if not, what other perception techniques should we be looking at?

Mike Kelly, a Ph.D. student, is approaching this by looking at people in a scene. An overall look at the scene with special purpose people-detection operators determines the areas of interest. A profile recognizer determines some gross parameters about the person. Using automatic pan, tilt, and lens selection we obtain a close-up view of the faces for further analysis.

Traditional edge detectors and region analyzers yield a great deal of spurious information. Therefore, we are using goal directed feature extraction procedures for the determination of location and shape eyes, nose, mouth, hair line etc. These procedures are not critically dependent on edge detection or thresholds.

In addition, we are investigating the use of color, texture, and environmental constraints in visual perception by machine. For any given lighting conditions one or more of these features can be expected to exhibit sharp discontinuities at region boundaries. During the next year we hope to solve the following specific problems concerning color and texture of a scene:

1) Using three color filters to obtain digitized images consisting of (toy) houses, cars, trees and rocks.

2) Develop operators which permit picture segmentation using a similarity or homogenity criterion based on their color parameters at each point.

3) Extension of the operators to use depth, light intensity and texture features.

Two dimensional Fourier transform of the whole scene has been suggested for obtaining texture parameters. Besides being computationally expensive, this cannot conveniently be used in conjunction with color, light intensity, and depth parameters. What is needed is the concept of local texture, i.e., texture parameters at a given point, so that one may decide whether that point belongs to one object or another in the scene.

Local texture at any point can be determined by Fourier transform of a small neighborhood around the point (perhaps convolved with a Bell-shaped or a Cone-shaped weighting function). This would still be computationally expensive. An alternative is to count the number of local maxima and minima (at a certain distance apart) in successive neighborhoods around the point. We hope to develop a local texture determination program and use it with 3-D operators to separate regions and objects.

We are currently completing a first version of our vision scheme which, to facilitate experimentation, allows human intervention at several stages in the scene analysis process. As intermediate stages of analysis are displayed, the user will be able to interrupt and add information to the system. Using this system and some hard thought, we hope to come up with a reasonable first cut at the multi-level vision system. We will then begin the process of refining this system and adding to its basic capabilities.

3.4 Arm Control

Work is also proceeding on improvements to the hand part of the hand-eye system. Along with improved mechanical design, we are generally concerned with the computer control of moving manipulators. This latter is an area presenting several challenging problems, including dynamics of the manipulator, path or space constraints, and the effects of computer control when the sampling time approaches the response time of the controlled element. Pieper approached the manipulator control problem kinematically [11], developing a method for iteratively solving the positioning for manipulators which are unsolvable analytically. In addition, he developed a method by which a manipulator may be moved from one position to another while avoiding certain obstacles. The path chosen does not account for the dynamics of the manipulator and in practice the manipulator is lead slowly along the path by the computer's control program.

Mike Kahn has approached the problem from the viewpoint of dynamics, obtaining a near-minimum time control for moving the manipulator from one position to another. The solution accounts for the dynamic reactions of one joint on another and the gravity loads on the manipulator. Certain linearization assumptions in the solution restict the applied torque-ratios on the manipulator to given ranges and also limit the validity of the solution to a range of motions in which the inertia changes seen by a given joint are not excessively large. The resulting control is bang-bang and the solution is in the form of switching curves for the control of each joint. The resulting path is a strictly dynamic one, in that it does not account for obstacles which may exist in the working space.

Several problems remain in the area of computer control of manipulators. 1) For each joint the reversal curve describes the point (dependent on both position and velocity) at which bang-bang deceleration must be applied to the joint so that the arm will arrive at the goal at rest. The problem is to develop a simplified approach to constructing reversal curves for each joint of the manipulator. A possible way of doing this would be to approximate the reversal curves by tangents at the expected switching points. In addition, a simplified method of dealing with large inertial changes seen by the joints would be desirable. 2) To combine the dynamical and kinematical viewpoints is obtaining a control scheme which accounts for both the dynamics of a rapidly moving manipulator and the path constraints imposed by obstacles in the workspace. This problem in multivariable optimal control with state-space constraints also has application in other areas of control theory. 3) An investigation of the influence of sampling rate and computation time on how well a fast manipulator can be controlled. An increase in the sophistication of a computer control scheme could allow an increase in manipulator speed relative to sampling rate.

3.5 Visual Control of a Vehicle

Included in the visual manipulation area is the development of a computer controlled vehicle. We have a battery powered cart with a TV link to the computer and a radio link back. Rod Schmidt, who adapted the hardware has also written a program that, under good seeing conditions, successfully follows a white line that is similar to those found on some roads. At present, programs are being written to analyze road scenes for the boundaries of the road, cars, obstacles such as dogs and holes, and other relevant features. We believe that within five years a computer-controlled vehicle can be brought to a pseudo-practical level; e.g., a computer controlled car could drive with low reliability from our lab to the Bayshore Freeway, paying proper attention to the road, other cars, pedestrians, animals, and traffic signals. A high reliability, reasonable cost system would then still require extensive development.

From a scientific point of view the computer driven vehicle presents several features of interest. First of all, the outputs required are quite simple: turning the wheels, and using the brakes and accelerator. Secondly, on a higher level the reaction to identified objects is simple, they are either to be ignored (shadows), avoided, or waited for. On the other hand, the class of objects that has to be dealt with is very large compared to that for manipulation problems. Many kinds of obstacles might be avoided without every identifying them precisely.

However, the advantages of working on computer controlled vehicles using visual information are not merely scientific. Once the basic problems of locating and identifying the relevant objects in a scene have been solved, technology will provide us with computers that can perform these tasks faster and more reliably than man. This will make possible a number of applications of both technological and economic importance.

There are, of course, major scientific problems to be solved before applications are feasible. The most apparent of these are the representation problem of scenes where obstacle avoidance is the goal and the consequent scene analysis problem under varying lighting conditions.

We are presently writing programs for locating major features such as the white line or dashed white line and the edge of the road. Our goals for the next year include finishing these programs and also recognizing some class of obstacles (such as cars). The programs will be checked by driving the cart around the building under various conditions.

REFERENCES

1. Earnest, L.D., "On Choosing an Eye for a Computer", Memo AI-51, Computer Science Dept., Stanford University, Stanford, California, 1967.

2. Ernst, H.A., "MH-1 a Computer Operated Mechanical Hand", Doctoral Thesis in Electrical Engineering, M.I.T., Cambridge, Massachusetts, 1961.

3. Feldman, J., and Rovner, P.D., "An Algol-based Associative Language", Memo AI-66, Computer Science Dept., Stanford University, Stanford, California, to appear in Comm. ACM, August 1969.

4. Gibson, J., The Senses Considered as Perceptual Systems, Houghton-Mifflin, Boston, 1966.

5. Guzman, A., Decomposition of Visual Scene into Three-dimensional Bodies", Proj. FJCC, 1968.

6. Guzman, A., "Some Aspects of Pattern Recognition by Computer", MAC-TR-37, Project MAC, M.I.T., Cambridge, Massachusetts, 1967.

7. Hueckel, M., "Locating Edges in Pictures", forthcoming A.I. Memo, Computer Science Dept., Stanford University, Stanford, California (to appear).

8. Nilsson, N., "The Stanford Research Institute Robot Project", Proc. Joint Int.Conference on Artificial Intelligence, Washington, D.C., 1969.

9. Paul, R., Falk, G., Feldman, J., "The Computer Representation of Simply Described Scenes", Proc. Illinois Graphics Conference, April 1969.

10. Paul, R., "A Representation of a Tower of Plano-convex Objects", Artificial Intelligence Laboratory, Operating Note No. 41, Stanford University, Stanford, California, August 1968.

11. Pieper, D., "The Kinematics of Manipulators under Computer Control" (Ph.D. Dissertation in Mechanical Engineering), Memo AI-73, Computer Science Dept., Stanford University, Stanford, California, 1968.

12. Pingle, K., "Hand-eye Library File", Artificial Intelligence Laboratory Operating Note No. 35, Stanford University, Stanford, California, August 1968.

13. Pingle, K., "A List Processing Language for Picture Processing", Artificial Intelligence Laboratory Operating Note No. 33, Stanford University, Stanford, California.

14. Pingle, K., Singer, J.A., and Wichman, W.M., "Computer Control of a Mechanical Arm Through Visual Input", Proc. IFIP Conference, Edinburgh, 1968.

15. Pohl, I., "Search Processes in Graphs", Stanford University Dissertation, forthcoming.

16. Roberts, L.G., "Machine Perception of Three-Dimensional Solids" in Optical and Electro-Optical Processing of Information, MIT Press, Cambridge, Massachusetts, 1965.

17. Samuel, A., "Studies in Machine Learning Using the Game of Checkers", IBM Journal, November 1967.

18. Shapiro, G., "Advanced Hand-Eye Manipulating", Internal Memo, Stanford Artificial Intelligence Project, Stanford University, Stanford, California.

19. Scheinman, V., "Design of a Computer Controlled Manipulator", Engineering Thesis in Mechanical Engineering, Stanford University, Stanford, California, 1969.

20. Sobel, I., forthcoming Electrical Engineering Thesis on "Visual Accommodation in Machine Perception", Stanford University, Stanford, California.

21. Swinehart, D., "Golgol III Reference Manual", Artificial Intelligence Laboratory Operating Note No. 48, Stanford University, Stanford, California.

22. Tenenbaum, J., "An Integrated Visual Processing System", forthcoming.

23. Wichman, W.H., "Use of Optical Feedback in the Computer Control of an Arm", Engineers Thesis, Stanford University, Stanford, California, August 1967.

24. McCarthy, J., Earnest, L.D., Reddy, D.R., Vicens, P.J., "A Computer with Hands, Eyes, and Ears", Proc. FJCC 1968.

4. Speech Recognition

Our research in speech recognition has been mainly concerned with developing systems for fast and accurate voice input to computers. At present the PDP-10 System can recognize limited sets of words, (the largest vocabulary tested is 560 words) phrases, and a few connected speech utterances made out of these. The time for recognition is around 2 to 15 seconds (depending on the size of the vocabulary) per word with an accuracy of 95 to 98 percent. The system has been used for voice control of a computer-controlled mechanical hand. Some relevant references are given at the end of this section.

Both the goals and methodologies of various ARPA supported speech projects are different and complementary to one another. Our effort differs from BBN's effort in that we are more interested in connected speech than limited vocabularies. BBN's research attempts to extract and use traditional phonological features in recognition. The speech analysis-synthesis project at University of California at Santa Barbara is directed more at vocoder-type applications with no attempt at machine recognition. UCSB's effort attempts to obtain an accurate and compact representation of speech parameters so that the original utterance can be effectively resynthesized from these parameters. Our recognition effort is based on easily extracted acoustic distinctive features (which may or may not be related to phonological distinctive features) and depends strongly on structural relationships of word-level and sentence-level syntax. Our system is designed not to be critically dependent on obtaining accurate parameters since it is usually difficult to obtain high quality speech in a computer environment. Whether we use crude or accurate representation of speech the main problem is the separation of the characteristics of the intended utterance from that of the speaker and the environment.

Future Plans

There are many outstanding unresolved questions in speech analysis and perception research. During the next few years we expect to concentrate on those problems which, when solved, should significantly enhance our knowledge about speech communication with machines, namely, normalization of speaker differences, languages for man-machine voice communication, and a better understanding of problems involved in carrying on speech conversations with machines. The following paragraphs outline our research plans in these directions.

1. Speaker Normalization. At present, we have to train the system with all the speakers. To recognize the speech of a random speaker, without explicitly training the system with each word or phrase he is likely to utter, requires some form of speaker identification and normalization. Previous attempts at speaker recognition have been disappointing. Our approach will involve having the speaker speak a kernel set of sentences initially and then every time he begins to use the machine he would identify himself by saying "I am John McCarthy" or some such. The system will then modify the recognition algorithm to correspond to his "acoustic signature".

2. Language Design for Man-Machine Voice Communication. A language for speaking to machines should not be unwieldy and ambiguous like English, nor should it be unnatural like a programming language. We believe a Lisp-like quasi-functional notation might be more convenient for some applications, i.e., it is more desirable to say "add Alpha to Beta" rather than "Beta Equals Alpha Plus Beta Semi-colon". The structure

of spoken machine languages will vary from task to task. We hope to have a system in which the user will specify the language as a BNF grammar or some such. The system will then analyze the grammar for possible phonemic, word boundary, and syntactic ambiguities and suggest possible modifications. The system will also provide a skeleton recognizer which the user can use to train the system to recognize his language for one or several speakers.

3. We hope to use Signature-Table learning techniques to achieve more accurate phoneme recognition, (see Section 5.1).

4. Conversational Systems. To be able to carry on speech conversation, a machine must not only have speech recognition and synthesis capabilities but also a good sematic model. At present we have a highly simplified conversational system which can recognize and respond to about 20 phrases using time-domain compressed speech as output, and based on a simple semantic model. The main purpose of this research will be to help us isolate and identify the main bottlenecks in achieving our goal, and to evaluate the performance of various recognition and synthesis models in an exacting task environment.

In the next five to ten years we expect to have a processor of the speed of a PDP-10 servicing 3 to 6 terminals with speech input and output capability. Such a system could be used as:

1. An extra motor process when hands and feet are already in use. (Possible applications: aeronautics, space and interactive graphics.)

2. A fast response, and fast data rate motor process (when a real time control of devices by a computer is performed under human supervision).

3. Natural mode of communication between man and machine. Some possibilities are day-to-day calculations (voice controlled desk calculator), information retrieval and question-answering systems, dictation (of programs,

letters, etc.), medical diagnosis, and psycho-therapy.

If, as it seems likely, a computer 5 to 10 times the speed of the PDP-10 becomes available for about the same cost as a PDP-10 in the next 5 to 10 years, then we believe it will be possible to extend the above system to service 30 to 50 terminals with voice input and output capability. Such a system should then be competitive with conventional input-output equipment.

REFERENCES

1. Reddy, D.R. and Robinson, A.E., "Phoneme to Grapheme Translation of English", IEEE Trans. Audio and Electroacoustics, AU-16, 2 240-246 (1968).

2. Reddy, D.R., "Digital Speech Compression in Time Domain", Presented at the 76th meeting of Acoustical Society of America, JASA, 44, 1, 391 (1968) (Abstract form only).

3. Reddy, D.R., "On Computer Transcription of Phonemic Symbols", J. Acoust. Soc. Am., 44, 2, 638-639 (1968) (letter).

4. Reddy, D.R., "Consonantal Clusters and Connected Speech Recognition", Proc. of the 6th International Congress on Acoustics, Tokyo (1968).

5. Vicens, P., "Preprocessing for Speech Analysis", Artificial Intelligence Memo AI-71, (October 1968).

6. Reddy, D.R., and Vicens, P., "A Procedure for Segmentation of Connected Speech", J. Audio Engr. Soc., 16, 4, 404-412, (1968).

7. McCarthy, J., Earnest, L.D., Reddy, D.R., and Vicens, P., "A Computer with Hands, Eyes and Ears", Proc. of the FJCC 1968, 329-337, (1968).

8. Reddy, D.R., "On the Use of Environmental, Syntactic and Probabilistic Constraints in Vision and Speech", Memo AI-78, (January 1969).

9. Reddy, D.R., and Neely, R.B., "Contextual Analysis of Phonemes of English", Memo AI-79, (January 1969).

10. Vicens, P., "Aspects of Speech Recognition by Computer", Memo AI-85, (April 1969).

5. Heuristic Search

5.1 Machine Learning

Enough progress has been made in the development of the new signature table procedure as a machine learning technique to justify a serious attempt to apply these methods to problems of practical import. At the same time there is still much to be learned about machine learning in general; enough, in fact, to warrant the continuation and even the expansion of basic work using the more formal context of a game. As has been said many times, the use of a game environment enables one to exclude the many extraneous and complicating details of real life situations and to concentrate one's entire attention on the basic problems under study.

The game of checkers still appears to offer many advantages as a study vehicle and we are accordingly continuing work on this game. The Samuel checker program has been completely rewritten for the PDP-10 and the playing portion is now thought to be reasonably bug free. The program is a good deal more complicated than was the IBM-7094 version and the playing portion quite successfully trades memory space for time so as to more than compensate for the difference in machine speeds. Even without benefit of a satisfactory amount of learning the program plays quite well. The learning aspect of the program is, however, still not entirely satisfactory as we have not yet been able to compensate for the difference in machine speeds in this portion of the program. As the program now stands, many hours of hard-to-come-by machine time with large amounts of core are required to test out each fairly minor modification in learning. Steps are being taken to alleviate this situation both by rewriting the well understood portions of the program to increase speed and by exercising extreme care in the design of experiments which will yield the greatest amount of information per hour of machine usage. There do appear to be some rather fundamental limitations as to what can be done along these lines and we may always be faced with the need for large amounts of machine time as long as we continue to experiment with different learning techniques.

Work is also continuing on the game of Go which offers another formal system but one in which the look-ahead procedure (so useful in checkers and chess) cannot usually be applied. This work is continuing at a slow pace with but one graduate student involved. The difficulties of the game and lack of knowledgeable players makes it hard to recruit people for this work. Go does, however, present a quite different environment from checkers and chess while still retaining the essential characteristic of formal simplicity and is therefore well worth study.

Speech recognition is engaging our attention as a very fruitful field in which to apply the signature table learning scheme developed by Samuel for checkers. Messrs. Samuel and Reddy are collaborating in this work. The general method of attack is to write a collection of carefully thought out general purpose subroutines which can be directly used in speech recognition but which will hopefully be useful for machine learning applications in quite disjoint fields of endeavor. Speech recognition was chosen largely because of the prior existence of significant work in this field in our laboratory. We now feel that this was a particularly happy choice in view of the fact that the signature table technique seems to be unusually well suited for this specific application.

Our initial efforts are in phoneme identification although we are also beginning to think about ways in which the same techniques can be brought to bear on the problems of segmentation and word recognition. Already several quite new ideas have emerged and we are extremely hopeful of success in this work.

5.2 Automatic Deduction

A theorem-proving program based on the Resolution Principle [1] for First Order Logic has been implemented using the PDP-10 LISP system. This program incorporates all of the efficiency strategies for proof search that are the subject of references [2,3,4,5,6], and has been used to compare the relative merits of various combinations of these strategies. In addition, it also has a special replacement rule for equality, [7]. The program is capable of dealing with all of the basic theorems of elementary algebra (Theory of Groups, Rings, Fields, Boolean Algebras and Modular Lattice Theory) and number theory that have been proved by programs discussed in the above references, and also with simple questions that arise in information retrieval, question-answering and automatic program writing [8]. A rudimentary interactive system has been incorporated into the program to allow the user to question and make suggestions to the program in the course of a proof search.

It is proposed to continue research in this area along both theoretical and practical lines towards the goal of developing this program into a practicable basis for automatic mathematics systems (proof checking and proposition testing), program writing, information retrieval, and other projects in Artificial Intelligence. Specific lines of research will include the following: (1) the further development of the interactive feature, (2) the addition of a proof-tree analyzer for extracting solutions to existential statements from proofs of them, (this is a basic part of the program writing and information retrieval applications), (3) the extension of the efficiency strategies mentioned above to a proof system incorporating rules of inference for the basic relations of equality, associativity, commutativity, etc., (4) the use of the algebraic symplification system, Reduce [9] in some of the phases of the proof search procedure, (5) the incorporation of decision procedures for certain classes of problem, (6) the extension of the basic logical system to a many-sorted calculus, and possibly the introduction of modal logic and w-order logic.

At present time, this is an area of intense endeavor and, using the facilities available here, we propose to test each theoretical possibility on practical problems.

REFERENCES

1. Robinson, J.A., "A Machine-Oriented Logic Based on the Resolution Principle", JACM, Vol. 12, No. 1, pp. 23-41, January 1965.

2. Luckham, D., "Some Tree-Paring Strategies for Theorem-Proving", Machine Intelligence 3 , D. Michie (ed), Edinburgh University Press, pp. 95-112, 1968.

3. Luckham, D., "Refinement Theorems in Resolution Theory", Memo AI-81, Computer Science Dept., Stanford University, Stanford, California, March 1969.

4. Wos, L., Carson, D., Robinson, G., "The Unit Preference Strategy in Theorem Proving", AFIPS Conf. Proc. 26, Washington, D.C., Spartan Books, pp. 615-621, 1964.

5. Wos, L., et. al., "Efficiency and Completeness of the Set of Support Strategy in Theorem Proving", JACM, Vol. 12, No. 4, pp. 536-541, October 1965.

6. Wos, L., et. al., "The Concept of Demodulation in Theorem-Proving", JACM, Vol. 14, No. 4, pp. 698-704.

7. Robinson, G., and Wos, L., "Paramodulation and Theorem-Proving in First-Order Theories with Equality", Machine Intelligence IV, D. Michie (ed), (to appear - 1969).

8. Green, C., "Theorem-Proving by Resolution as a Basis for Question-Answering Systems", Machine Intelligence 4, D. Michie (ed), American Elsevier, New York, 1969.

9. Hearn, A.C., "Reduce Users Manual", Memo AI-50, Computer Science Dept., Stanford University, Stanford, California 1968.

6. Models of Cognitive Processes

6.1. The Heuristic Dendral Project

The Heuristic DENDRAL Project explores the processes of problem solving, hypothesis formation, and theory formation in scientific work. The particular task chosen for the exploration initially was the determination of the molecular structure of organic molecules from mass spectral data, but is now being extended to include other spectral data.

In the interpretation of mass spectra of organic molecules, a chemist seeks a hypothesis about molecular structure that will serve to explain given empirical data produced by an instrument called a mass spectrometer. Heuristic DENDRAL is a LISP program that performs in this task environment. The overall structure of the program is a "scientific method" cycle, consisting of phases of preliminary inference making, systematic hypothesis generation, prediction from hypotheses, and test against empirical data with evaluation of goodness-of-fit.

The problem solving behavior of the program is remarkably good for the limited subset of organic molecules with which we have worked, in a task that is not a "toy" problem but a "real life" problem of considerable interest to science. For these molecules, the program's ability is at least equal to, and usually better than, that of professional mass spectrum analysts.

The project personnel are: Professor Edward Feigenbaum of the Computer Science Department and Professor Joshua Lederberg of the Department of Genetics; Dr. Bruce Buchanan, Instructor of Genetics and Computer Science; Georgia Sutherland, Research Associate in Computer Science; and Al Delfino, staff programmer. Affiliated with the project from the Chemistry Department are Professor Carl Djerassi, Professor of Chemistry, and Dr. Alan Duffield and Dr. Gustav Schroll of his Mass Spectrometry Laboratory. Others affiliated with the project are: Professor Malcolm Bersohn, Chemistry Department, University of Toronto, visiting on sabbatical; Mr. Dennis Brown, Computer Science Department student NSF Fellow, and Mr. Isu Fang, Computer Science Department student AID Fellow.

Eleven scientific reports describing the progress of the project have been published or are in press. The most complete and detailed description of the Heuristic DENDRAL program and its implementation was published in the book Machine Intelligence 4 (7). The most concise description was included as part of Feigenbaum's artificial intelligence survey for the IFIP68 Congress in Edinburgh (8). Results that deal specifically with isomers of organic molecules and with mass spectrometry of simple chemical groups have been submitted to (3), and accepted for (1,2), the Journal of the American Chemical Society. An early discussion of Heuristic DENDRAL from the point of view of the modeling of induction and heuristic rules of judgment in scientific problem solving was published in the book Formal Representations for Human Judgment (9). Heuristic DENDRAL is, of course, a precisely specified and working model of induction and empirical inquiry in a scientific task. As such, it is of interest to Philosophy of Science, which has always had as its main concern an understanding of the nature of scientific inquiry. This interest is addressed in an article on Heuristic DENDRAL to appear in the British Journal for the Philosophy of Science (5). Lederberg's DENDRAL algorithm and his DENDRAL structure notation form the basis for the heuristic problem solving system, and are described in various publications that address themselves to the topology of organic molecules and the manipulation of these organic graphs by computer (6, 10, 11, A).

Chemists supported by other grants and fellowships will continue to be working with the project in the next period as they did in the last period.

Work to be Undertaken During the Period of the Proposal:

1. Application of A.I. to Chemical Inference

As an application to chemistry, Heuristic DENDRAL is open-ended (at least in relation to mass spectrum analysis and some closely allied problems). Increments of new chemical knowledge, formalized by the man-machine technique we have been using, produce beneficial increments in the problem solving capability of the program. This type of work -- of broadening and deepening the application to chemistry -- will continue, with the assistance of our chemist affiliates. This will include greater concentration on ringed structures; adding many more organic "functional groups" to the analysis; bringing to bear additional chemical data (particularly infrared, ultra-violet, and N.M.R. spectra); improving the theory of molecular fragmentation used in the Predictor; and improving the goodness-of-fit evaluation criteria. We will also attempt to make the program available to the rest of the scientific community in LISP on the IBM 360.

2. The Problem of Representation for Problem Solving Systems

The problem of representation has been much discussed as a key problem of artificial intelligence research (8). We will be studying this problem in a number of different ways in connection with the various representations of the theory of fragmentation of organic molecules that are currently used by the Heuristic DENDRAL program. We will be attempting to write programs that will automatically alter the representation of chemical knowledge, from forms ill-adapted to a particular type of problem solving in the system (but perhaps more convenient for the input of knowledge by chemists) to forms more efficient in the problem solving. Initial experiments will involve programs that will move knowledge about fragmentation processes from the Predictor's form to the Preliminary Inference Maker's form; and programs that will re-represent the Predictor itself in a more convenient flexible form.

3. The Problem of Knowledge Assimilation

This problem is closely allied to the problem of representation. The Heuristic DENDRAL program has its knowledge (its "model") of mass spectrometric and other chemical processes represented in at least four different forms, each particularly well-suited to the performance of a specific type of inference process. But how can the system check the consistency of these various forms of the same knowledge? When a change is made to one of the representations, involving the input of new chemical knowledge at that point, can it be guaranteed that the proper changes will be made in the other representations to preserve consistency in the system's knowledge of chemistry? We have a way of attacking this problem that involves some of the experience to be gained in studies of the problem of representation discussed above, and we plan to devote considerable effort in this period to working this out.

4. The Problem of Man-Machine Communication for Transfer of Knowledge

In the DENDRAL system, man-machine interaction is used not merely to debut and run the program but also as the primary vehicle for the communication of new knowledge about chemistry to the program. The problem of how to do this effectively is intimately bound up with the problems of representation and knowledge assimilation. Chemists do not necessarily (or even usually) have their knowledge about the chemical world represented in the fashion most convenient for our various programs to use. Thus, automatic re-representation of knowledge from the "ground state" forms used by our chemists to more "finely tuned" forms for use by the program is essential.

The work on knowledge assimilation will allow us to experiment with more intelligent man-machine communication. In addition, to facilitate the transfer of knowledge via the man-machine interaction, we plan to develop a "problem-oriented" language for chemistry, oriented toward the problem of expressing chemical knowledge to the system and translating this knowledge to the appropriate internal representation.

5. The Problem of Efficient Design

How knowledge of the chemical world is employed by DENDRAL at the various points in its "scientific method" cycle effects markedly the efficiency of the problem solving process. Thus, some important questions of systems design are raised, which must be carefully studied and understood -- a task that we propose to do. Since there is no epistemological theory which can serve as a guide, this understanding must be gained experimentally. For example, under what general conditions does it "pay" to apply knowledge in the DENDRAL hypothesis generation stage rather than in the hypothesis validation stage (and vice-versa)? Can, indeed, any general answers be given to system design questions of this sort?

Summary and Generalization:

A problem solving program has been written which exhibits a high level of performance on a complex scientific inference task. Using this performance system as a springboard, a series of additonal program-writing endeavors, addressing themselves to what we believe are key problems in artificial intelligence research, are proposed. These involve the problems of: representation; knowledge assimilation; man-machine communications for transfer of knowledge; and efficient systems design. Though these problems are to be studied concretely in the context of chemical inference, they are clearly of considerable general interest. All of them, for example, are important in the development of computer programs for the intelligent "management" of large data bases. The key questions being asked involve how to organize and "manage" knowledge: how to do this most efficiently to control search, and how to get programs to do this automatically, in an effective manner, with minimal human intervention.

Project Publications (and publications in press):

1. Lederberg, J., Sutherland, G.L., Buchanan, B.G., Feigenbaum, E.A., Robertson, A.V., Duffield, A.M., and Djerassi, C., "Applications for Artificial Intelligence for Chemical Inference I. The Number of Possible Organic Compounds: Acyclic Structures Containing C,H,O and N". Journal of the American Chemical Society May 1969.

2. Duffield, A.M., Robertson, A.V., Djerassi, C., Buchanan, B.G., Sutherland, G.L., Feigenbaum, E.A., and Lederberg, J., "Applications of Artificial Intelligence for Chemical Inference II. Interpretation of Low Resolution Mass Spectra of Ketones". Journal of the American Chemical Society May 1969.

3. Schroll, G., Duffield, A.M., Djerassi, C., Buchanan, B.G., Sutherland, G.L., Feigenbaum, E.A., and Lederberg, J., "Applications of Artificial Intelligence for Chemical Inference III. Aliphatic Ethers Diagnosed by Their Low Resolution Mass Spectra and NMR Data". Submitted to the Journal of the American Chemical Society.

4. Sutherland, G.L., Heuristic DENDRAL: "A Family of LISP Programs". To appear in D. Bobrow (ed.), LISP Applications. Also, Stanford Artificial Intelligence Project Memo AI-80.

5. Churchman, C.W. and Buchanan, B.G., "On the Design of Inductive Systems: Some Philosophical Problems". British Journal for the Philosophy of Science, to appear Autumn 1969 (in press).

6. Lederberg, J., "Topology of Molecules". In The Mathematical Sciences, published for the National Academy of Sciences--National Research Council by MIT Press, Cambridge, 1969, pp. 37-51.

7. Buchanan, B.G., Sutherland, G.L., and Feigenbaum, E.A., "Heuristic DENDRAL: A Program for Generating Explanatory Hypotheses in Organic Chemistry". In D. Michie (ed), Machine Intelligence 4 University of Edinburgh Press, 1969. (Also, Stanford Artificial Intelligence Project Memo AI-62.)

8. Feigenbaum, E.A., "Artificial Intelligence: Themes in the Sound Decade". In Final Supplement to Proceedings of the IFIP68 International Congress, Edinburgh, August 1968. (Also Artificial Intelligence Project Memo AI-67.)

9. Lederberg, J., and Feigenbaum, E.A., "Mechanization of Inductive Inference in Organic Chemistry". In B. Kleinmuntz (ed) Formal Representations for Human Judgment, Wiley, 1968. (Also Stanford Artificial Intelligence Project Memo AI-54.)

10. Lederberg, J., "Hamilton Circuits of Convex Trivalent Polyhedra". American Mathematical Monthly 74, 522 (1967).

11. Lederberg, J., "Topological Mapping of Organic Molecules", Proceedings of the National Academy of Science, U.S. 53, 134 (1965).

REPORTS

A. Lederberg J. DENDRAL 64 - A system for computer construction, enumeration and notation of organic molecules as tree structures and cyclic graphs.

A.1. Part I, Notational algorithm for tree structures, NASA CR-57029 and STAR N65-13158.

A.2. Part II, Topology of cyclic graphs, NASA CR-68898 and STAR N66-14074.

A.3. Part III, A general outline of the DENDRAL system. Systematics of organic molecules, graph topology and Hamilton circuits, NASA CR-68899 and STAR N66-14075.

B. Sutherland, G.L., "A Computer Program for Generating and Filtering Chemical Structures. Stanford Artificial Intelligence Project Memo AI-49.

6.2 Language Research

There are a number of problems relating to the automatic processing of natural language which are continuing to be of interest. This is divided into three major sub-areas: associative data structures, models of cognitive structures and grammatical inference.

We have been studying the problems of associative memory in conventional computers for several years. The most recent development is the abiility to have several independent, parallel programs all sharing the same associative structure. We hope to study the problems of controlling access to a global structure in the context of hand-eye tasks. Another important tope to be studied is the addition of deductive inference cap bilities to the associative retrieval mechanisms.

One of the most interesting and difficult problems in artificial intelligence is the modeling of human cognitive structures. We have developed such a model [2] and are studying several problems in language processing and understanding with this model. The model is unique in that it uses the notion of consequence (temporal, causal, etc.) as a central element. We are developing theories of analogy, generalization over instances and the relation between perception and understanding using this model.

Work on grammatical inference continues to be fruitful. The theoretical work on decidability is complete [3] and we are looking at questions of optimal learning and teaching strategies. Many of these results will be directly converted into program heuristics. We are also studying the extension of these techniques to other problems of generalization.

REFERENCES

1. Feldman, J., and Rovner, P., "An Algol-Based Associative Language", Stanford Artificial Intelligence Memo AI-66, Stanford University, Stanford, California, August 1968.

2. Feldman, J.A., "First Thoughts on Grammatical Inference", Stanford Artificial Intelligence Memo AI-55, Stanford University, Stanford, California, August 1967.

3. Feldman, J.A., Gips, J., Horning, J., and Reder, S., "Grammatical Inference and Complexity", Stanford Artificial Intelligence Memo AI-89, Stanford University, Stanford, California, June 1969.

4. Becker, J.D., "The Modeling of Simple Analogic and Inductive Processes In a Semantic Memory System", Stanford Artificial Intelligence Memo AI-77, Stanford University, Stanford, California, January 1969.

6.3 <u>Higher Mental Functions</u>

The Higher Mental Functions Project is an affiliated project under the direction of Dr. Kenneth Mark Colby, who is supported by N.I.H, as a Research Career Scientist. This project is working on two problem areas: (1) The Study of credibility functions in humans as well as in aritificial systems, (2) Question-asking or interviewing programs which operate in natural language.

In the first area of interest the problem is to understand how a system, living or artificial, judges the credibility of new information based on information it already possesses. Descriptions of work already done on this problem can be found in Reference (1, 2, 3). Because the relevant variables are difficult to control in humans, it is considered necessary to develop artificial belief systems in which the belief processes under study can be brought under maximum control.

In the second domain of interest, the problem consists of machine understanding of natural language. We have have considerable experience in this area (See Reference 4) and we are currently developing a program which conceptually analyzes natural language input. (See Reference 5). Once the input is "understood" the program can generate questions based on information it has received thus far rather than being limited to a fixed set of questions.

REFERENCES

1. Colby, K.M., "Computer Simulation of Change in Personal Belief Systems", Behavorial Science, 12, 248-253 (1967).

2. Tesler, L., Colby, K.M., and Enea, H., "A Directed Graph for Computer Simulation of Belief Systems", Mathematical Biosciences, 2, 19-40 (1968).

3. Colby, K.M., Tesler, L., Enea, H., "Search Experiments with the Data Base of Human Belief Structure", Proc. International Joint Conference on Artificial Intelligence, Washington, D.C., May 1969.

4. Colby, K.M., and Enea, H., "Heuristic Methods for Computer Understanding of Natural Language in Context-Restricted On-Line Dialogues", Mathematical Biosciences, 1, 1-25, 1967.

7. Budget

$1,975,859 is needed to support the research program described above for the eighteen month period beginning 1 January 1969. The budget below is divided into an initial six month period ($613,527 from 1 January through 30 June 1970) and following twelve month period ($1,362,332 from 1 July 1970 through 30 June 1971). Separate budgets are given for the Heuristic Dentral Project (H.D.) and the other artificial intelligence projects (A.I.) in each period. The Higher Mental Functions Project is separately supported so no funds are needed for it.

The bulk of the funds requested are for salaries and personnel support costs. $20,000 per year is needed for test equipment, mostly in support of the visual perception and control projects. $80,000 is budgeted in the second period for additional displays consoles. The existing 6 console display system is currently in saturated use about 18 hours per day and the utilization is increasing.

BUDGET SUMMARY

For Continuation of SD 183

1 Jan 1970 - 30 Jun 1971

Budget Item	1 Jan 1970 - 30 Jun 1970		1 Jul 1970 - 30 Jun 1971		Total
	A.I.	H.D.	A.I.	H.D.	
Salaries	$240,483	$31,588	$509,825	$66,967	$848,863
Staff Benefits	29,579	3,885	62,708	8,237	104,409
University Overhead	137,075	18,005	290,600	38,171	483,851
Travel	8,900	2,250	17,800	4,500	33,450
Capital Equipment	10,000		100,000		110,000
Equipment Rental	28,176	2,520	56,352	5,040	92,088
Equipment Maintenance	25,000		50,000		75,000
Computer Time	2,000	36,000	4,000	72,000	114,000
Communications	7,500	1,000	15,000	2,000	25,500
Publications Costs	4,125	441	8,250	882	13,698
Other Operating Expenses	22,250	2,750	44,500	5,500	75,000
Subtotal by Project	$515,088	$98,439	$1,159,035	$203,297	$1,975,859
Total by Period	$613,527		$1,362,332		$1,975,859

BUDGET FOR CONTINUATION (1 Jan 70 - 30 Jun 71)

SD 183

I. ARTIFICIAL INTELLIGENCE	1 Jan 70 - 30 Jun 70	1 Jul 70 - 30 Jun 71
Faculty		
Adams, J.L., Assoc. Prof. of Mech. Engr., 1/6 time acad, yr., 1/2 time summer	$2,216	$4,698
Feldman, J., Assoc. Prof. of Computer Science, 1/2 time acad. yr., full time summer	5,391	11,429
Floyd, R., Assoc. Prof. of Computer Science, 1/2 time acad. yr., full time summer	7,291	15,457
Knuth, D., Prof. of Computer Science	—	
Manna, Z., Assist. Prof. of Computer Science, 1/2 time acad. yr., full time summer	4,628	9,811
McCarthy, J., Prof. of Computer Science, Principal Investigator, 1/2 time acad. yr., full time summer	8,333	17,666
Reddy, D.R., Assist. Prof. of Computer Science, 1/2 time acad. yr., full time summer	5,168	10,956
TOTAL FACULTY SALARIES	$33,027	$70,017
Research Staff		
Ashcroft, E.A. Research Assoc.	$ 6,600	$13,992
Baumgart, B., Systems Programmer	4,800	10,176
Beauchamp, J. Research Assoc.	6,300	13,356

Research Staff cont'd.

	1 Jan 70 - 30 Jun 70	1 Jul 70 - 30 Jun 71
Earnest, L., Research Assoc., Executive Officer	$11,500	$24,380
Feldman, G., Research Programmer 1/2 time acad. yr., full time summer	3,694	7,831
Gleason, G., Computer Systems Engineer	6,630	14,056
Grape, G., Research Programmer	5,400	11,448
Hueckel, M., Research Assoc.	6,600	13,992
Kay, A., Research Assoc. 3/4 time	6,000	12,720
Luckham, D. Research Assoc.	8,010	16,981
McGuire,E., Systems Programmer	4,800	10,176
Moorer, J., Systems Programmer	5,160	10,939
Paul, R., Research Programmer	6,780	14,374
Pingle, K., Research Programmer	5,940	12,593
Poole, D., Systems Programmer	5,550	11,766
Samuel, A., Senior Research Assoc., 3/4 time	10,000	21,200
Singer, J., Systems Programmer, Group Leader	7,650	16,218
Smith, D., Systems Programmer 1/5 time	1,080	2,290
Sproul, R., Systems Programmer	4,800	10,176
Weiher, W., Systems Programmer	5,400	11,448
TOTAL RESEARCH STAFF	$122,694	$260,112

		1 Jan 70 - 30 Jun 70	1 Jul 70 - 30 Jun 71
	Student Research Assistants (20)		
	J. Allen, R. Bajcsyova, J. Becker, J. Buchanan, L. Erman, G. Falk, R. Goodman, M. Kelly, R. Neely, P. Petit, L. Quam, J. Ryder, R. Schmidt, I. Sobel, D. Swinehart, J. Tenebaum, plus 4 unnamed:		
	TOTAL STUDENT RESEARCH ASSISTANTS	$47,000	$99,640
	Other Staff		
	Baur, Q., Secretary, 1/2 time	1,500	3,180
	Down, K., Research Coordinator, 9/10 time	5,400	11,448
	Panofsky, E., Electronics Technician	3,900	8,268
	Roark, D., Secretary	3,000	6,360
	Zingheim, T., Electronics Technician	4,350	9,222
	-----, Electronics Technician	3,600	7,632
	Semi-monthly Technicians	2,400	5,088
	TOTAL OTHER STAFF	$24,150	$51,198
	SUBTOTAL A.I. PROJECT SALARIES	$226,871	$480,967
	Allowance for 6% Salary Increase	13,612	28,858
	TOTAL A.I. PROJECT SALARIES	$240,483	$509,825
II.	Staff Benefits (12.3% Provisional through 31 August 1970)	$29,579	$62,708
III.	University Overhead (57%)	$137,075	$290,600
IV.	Travel		

2 foreign trips, $1,200 ea.	$2,400
9 trips east, $450 ea.	4,050
2 professional staff moves to Stanford, $750 ea.	1,500
Local travel	950
	$8,900

		1 Jan 70 - 30 Jun 70	1 Jul 70 - 30 Jun 71
IV. Travel (continued)			
5 foreign trips, $1,200 ea.	$6,000		
16 trips east, $450 ea.	7,200		
4 professional staff moves to Stanford, $750 ea.	3,000		
Local travel	1,600		
			$17,800
V. Capital Equipment			
Test Equipment (Oscilloscopes, arm and camera instrumentation, misc.)		$10,000	$20,000
Display generator and 6 display units (Similar to Data Disc Units)			80,000
TOTAL CAPITAL EQUIPMENT		$10,000	$100,000
VI. Equipment Rental			
IBM Disc File and Packs		$28,176	$56,352
VII. Equipment Maintenance (Based on past experience)		$25,000	$50,000
VIII. Computer Time (IBM 360-67)			
3 hours day rate, $500/hr	$1,500		
Supporting Services	500		
		$2,000	
6 hours day rate, $500/hr	$3,000		
Supporting Services	1,000		
			$4,000
IX. Communications (Telephones, data-phones, teletype)		$7,500	$15,000
X. Publications Costs		$4,125	$8,250
XI. Other Operating Expenses		$22,250	$44,500
TOTAL ARTIFICIAL INTELLIGENCE		$515,088	$1,159,035

		1 Jan 70 - 30 Jun 70	1 Jul 70 - 30 Jun 71
XII.	HEURISTIC DENDRAL		
	Faculty		
	Feigenbaum, E., Prof. of Computer Science, 65 % time acad. yr., full time summer	$8,500	$18,020
	Lederberg, J., Prof. of Genetics, 5 % time	1,000	2,120
	TOTAL FACULTY SALARIES	$9,500	$20,140
	Research Staff		
	Buchanan, B., Research Associate	$6,800	$14,416
	Delfino, A., Research Programmer	6,000	12,720
	Sutherland, G., Research Associate 2/3 time	3,750	7,950
	Brown, D., Student Research Asst., 1/2 time acad. yr., full time summer	1,500	3,180
	Semimonthly wages (undergraduate student)	750	1,590
	TOTAL RESEARCH STAFF	$18,800	$39,856
	Other Staff		
	-----, Secretary, 1/2 time	$1,500	$3,180
	SUBTOTAL HEURISTIC DENDRAL SALARIES	$29,800	$63,176
	Allowance for 6% Salary Increase	$1,788	$3,791
	TOTAL HEURISTIC DENDRAL SALARIES	$31,588	$66,967
XIII.	Staff Benefits (12.3% Provisional through 31 August 1970)	$3,885	$8,237
XIV.	University Overhead (57%)	$18,005	$38,171

			1 Jan 70 - 30 Jun 70	1 Jul 70 - 30 Jun 71
XV.	Travel			
	1 foreign trip, $1,200 ea.	$1,200		
	2 trips east, $450 ea.	900		
	Local travel	150		
			$2,250	
	2 foreign trips, $1,200 ea.	$2,400		
	4 trips east, $450 ea.	1,800		
	Local travel	300		
				$4,500
XVI.	Equipment Rental (Wylbur Terminals)		$2,520	$5,040
XVII.	Computer Time (IBM 360-67)			
	60 hours day rate, $500/hr	$30,000		
	Supporting Services	6,000		
			$36,000	
	120 hours day rate, $500/hr	$60,000		
	Supporting Services	12,000		
				$72,000
XVIII.	Communications (Telephones, data-phones, teletype)		$1,000	$2,000
XIX.	Publications Costs		$441	$882
XX.	Other Operating Expenses		$2,750	$5,500
	TOTAL HEURISTIC DENDRAL		$98,439	$203,297

XXI. Associated Groups - Non-Direct Support

A. Higher Mental Functions

Colby, K., Principal Investigator, Senior Research Associate

Hilf, F., Research Associate

Schank, R., Programmer

Tesler, L., Programmer

Smith, D., Research Programmer, 4/5 time

A. <u>Higher Mental Functions</u> (Continued)

Weber, S., Student Research Assistant, 1/2 time acad. yr., full time summer

Down, K., Research Coordinator, 1/10 time

Baur, Q., Secretary, 1/2 time

8. Cognizant Personnel:

For contractual matters, including overhead and patent questions:

Elwood C. Pierce
Office of the Research Administrator
Stanford University
Stanford, California 94305

For technical and scientific matters:

Professor John McCarthy, Principal Investigator
Professor Edward Feigenbaum, Associate Investigator
Dr. Arthur Samuel, Associate Investigator
Mr. Lester Earnest, Executive Officer
Computer Science Department
Stanford University
Stanford, California 94305

Telephone (415) 321-2300, extension 4971

For administrative matters, including questions relating to the budget, property acquisition and handling, etc:

Mr. Lester Earnest, Executive Officer
Mr. Kenneth Down, Research Coordinator
Computer Science Department
Stanford University
Stanford, California 94305

Telephone (415) 321-2300, extension 4971

APPENDIX A

PUBLICATIONS OF PROJECT MEMBERS

Articles and books by members of the Stanford Artificial Intelligence Project are listed here by year. Only publications subsequent to the individual's affiliation with the Project are given.

1963

1. J. McCarthy, "A Basis for a Mathematical Theory of Computation", in P. Biaffort and D. Hershberg (eds), Computer Programming and Formal Systems, North-Holland, Amsterdam 1963.

2. J. McCarthy, "Towards a Mathematical Theory of Computation" in Proc. IFIP Congress 62, North-Holland, Amsterdam, 1963.

3. J. McCarthy (with S. Boilen, E. Fredkin, and J.C.R. Licklider), "A Time-Sharing Debugging System for a Small Computer" in Proc. AFIPS Conf. (SJCC), Vol. 23, 1963.

4. J. McCarthy (with F. Corbato and M. Daggett), "The Linking Segment Subprogram Language and Linking Loader Programming Languages", Comm. ACM, July 1963.

1965

1. J. McCarthy, "Problems in the Theory of Computation", in Proc. IFIP Congress 65, Spartan, Washington, D.C., 1965.

1966

1. A. Hearn, "Computation of Algebraic Properties of Elementary Particle Reactions Using a Digital Computer", Comm. ACM, 9, pp. 573-577, August 1966.

2. J. McCarthy, "A Formal Description of a Subset of Algol" in T. Steele (ed), Formal Language Description Languages, North-Holland, Amsterdam, 1966.

3. J. McCarthy, "Information", Scientific American, September 1966.

4. D. Reddy, "Segmentation of Speech Sounds", J. Acoust. Soc. Amer., August 1966.

1967

1. S. Brodsky and J. Sullivan, "W-Boson Contribution to the Anomalous Magnetic Moment of the Muon", Phys Rev 156, 1644, 1967.

2. J. Campbell, "Algebraic Computation of Radiative Corrections for Electron-Proton Scattering", Nuclear Physics, Vol. B1, pp. 238-300, 1967.

3. E. Feigenbaum, "Information Processing and Memory" in Proc. Fifth Berkeley Symposium on Mathematical Statistics and Probability, Vol. 4, U.C. Press, Berkeley, 1967.

4. J. Goodman, "Digital Image Formation from Electronically Detected Holograms", in Proc. SPIE Seminar on Digital Imaging Techniques, Soc. Photo-Optical Instrumentation Engineering, Redondo Beach, California, 1967.

5. J. Goodman, "Digital Image Formation from Electronically Detected Holograms", Applied Physics Letters, 1 August 1967.

6. A. Hearn, "REDUCE, A User-Oriented Interactive System for Algebraic Simplification, Proc. ACM Symposium on Interactive Systems for Experimental Applied Mathematics, August 1967.

7. J. Lederberg, "Hamilton Circuits of Convex Trivalent Polyhedra", American Mathematical Monthly 74, 522, 1967.

8. J. McCarthy, D. Brian, G. Feldman, and J. Allen, "THOR - A Display Based Time Sharing System", AFIPS Conf. Proc., Vol. 30, (FJCC), Thompson, Washington, D.C., 1967.

9. J. McCarthy, "Computer Control of a Hand and Eye", in Proc. Third All-Union Conference on Automatic Control (Technical Cybernetics), Nauka, Moscow, 1967 (Russian).

10. D. Reddy, "Phoneme Grouping for Speech Recognition", J. Acoust. Soc. Amer., May 1967.

11. D. Reddy, "Pitch Period Determination of Speech Sounds", Comm. ACM, June 1967.

12. D. Reddy, Computer Recognition of Connected Speech", J. Acoust. Soc. Amer., August 1967.

13. A. Samuel, "Studies in Machine Learning Using the Game of Checkers, II-Recent Progress", IBM Journal, November 1967.

14. G. Sutherland (with G.W. Evans and G.F. Wallace), Simulation Using Digital Computers, Prentice-Hall, Engelwood Cliffs, N.J., 1967.

<u>1968</u>

1. E. Feigenbaum, J. Lederberg and B. Buchanan, "Heuristic Dendral", <u>Proc. International Conference on System Sciences</u>, University of Hawaii and IEEE, University of Hawaii Press, 1968.

2. E. Feigenbaum, "Artificial Intelligence: Themes in the Second Decade", <u>Proc. IFIP Congress 1968</u>.

3. J. Feldman (with D. Gries), "Translator Writing Systems", <u>Comm. ACM</u>, February 1968.

4. J. Feldman (with P. Rovner), "The Leap Language Data Structure", <u>Proc. IFIP Congress</u> 1968.

5. R. Gruen and W. Weiher, "Rapid Program Generation", <u>Proc. DECUS Symposium</u>, Fall 1968.

6. A. Hearn, "The Problem of Substitution", <u>Proc. IBM Summer Institute on Symbolic Mathematics by Computer</u>, July 1968.

7. D. Kaplan, "Some Completeness Results in the Mathematical Theory of Computation", <u>ACM Journal</u>, January 1968.

8. J. Lederberg and E. Feigenbaum, "Mechanization of Inductive Inference in Organic Chemistry", in B. Kleinmuntz (ed.), <u>Formal Representation of Human Judgment</u>, John Wiley, New York, 1968.

9. J. McCarthy, "Programs with Common Sense" in M. Minsky (ed.), <u>Semantic Information Processing</u>, MIT Press, Cambridge, 1968.

10. J. McCarthy, L. Earnest, D. Reddy, and P. Vicens, "A Computer with Hands, Eyes, and Ears", <u>Proc. AFIPS Conf.</u> (FJCC), 1968.

11. K. Pingle, J. Singer, and W. Wichman, "Computer Control of a Mechanical Arm through Visual Input", <u>Proc. IFIP Congress 68</u>, 1968.

12. D. Reddy, and Ann Robinson, "Phoneme-to-Grapheme Translation of English", <u>IEEE Trans. Audio and Electroacoustics</u>, June 1968.

13. D. Reddy, "Computer Transcription of Phonemic Symbols", <u>J. Acoust. Soc. Amer</u>., August 1968

14. D. Reddy, and P. Vicens, "Procedure for Segmentation of Connected Speech", J. Audio Eng. Soc., October 1968.

15. D. Reddy, "Consonantal Clustering and Connected Speech Recognition", <u>Proc. Sixth International Congress on Acoustics</u>, Vol. 2, pp. C-57 to C-60, Tokyo, 1968.

1968 (cont.)

16. A. Silvestri and J. Goodman, "Digital Reconstruction of Holographic Images", '68, NEREM Record, IEEE, Vol. 10, pp. 118-119, 1968.

17. L. Tesler, H. Enea, and K. Colby, "A Directed Graph Representation for Computer Simulation of Belief Systems", Math. Bio. 2, 1968.

<u>1969 (to date)</u>

1. J. Beauchamp (with H. Von Foerster) (eds), <u>Music by Computers</u>, John Wiley, New York, 1969.

2. J. Becker, "The Modeling of Simple Analogic and Inductive Processes in a Semantic Memory System," <u>Proc. International Conf. on Artificial Intelligence</u>, Washington, D.C., 1969.

3. B. Buchanan and G. Sutherland, "Heuristic Dendral: A Program for Generating Hypotheses in Organic Chemistry", in D. Michie (ed.), <u>Machine Intelligence 4</u>, American Elsevier, New York, 1969.

4. B. Buchanan (with C. Churchman), "On the Design of Inductive Systems: Some Philosophical Problems", British Journal for the Philosophy of Science, Autumn 1969 (in press).

5. K. Colby, L. Tesler, and H. Enea, "Experiments with a Search Algorithm for the Data Base of a Human Belief System", <u>Proc</u>. International Conference on Artificial Intelligence, Washington, D.C., 1969.

6. K. Colby and D.C. Smith, "Dialogues between Humans and Artificial Belief Systems", <u>Proc. International Conference on Artificial Intelligence</u>, Washington, D.C., 1969.

7. A. Duffield, A. Robertson, C. Djerassi, B. Buchanan, G. Sutherland, E. Feigenbaum, and J. Lederberg, "Application of Artificial Intelligence for Chemical Inference II. Interpretation of Low Resolution Mass Spectra of Ketones", J. of American Chemical Society, May 1969.

8. J. Feldman, G. Feldman, G. Falk, G. Grape, J. Pearlman, I. Sobel, and J. Tenenbaum, "The Stanford Hand-Eye Project", <u>Proc.International Conf. on Artificial Intelligence</u>, Washington, D.C., 1969.

9. J. Feldman (with P. Rovner), "An Algol-based Associative Language", <u>Comm. ACM</u>, August 1969.

10. T. Ito, "Note on a Class of Statistical Recognition Functions", <u>IEEE Trans. Computers</u>, January 1969.

11. J. Lederberg, "Topology of Organic Molecules", National Academy of Science, <u>The Mathematical Sciences: a Collection of Essays</u>, MIT Press, Cambridge 1969.

12. J. Lederberg, G. Sutherland, B. Buchanan, E. Feigenbaum, A. Robertson, A. Duffield, and C. Djerassi, "Applications of Artificial Intelligence for Chemical Inference I. The Number of Possible Organic Compounds: Acyclic Structures Containing C,H,O, and N", <u>J. Amer. Chem. Soc.</u>, May 1969.

1969 (cont.)

13. D. Luckham, "Refinement Theorems in Resolution Theory", Proc. 1968 IRIA Symposium in Automatic Deduction, Versailles, France, (in press).

14. Zohar Manna, "Properties of Programs and the First Order Predicate Calculus", J. ACM, April 1969.

15. Zohar Manna, "Formalization of Properties of Programs", J. System and Computer Sciences, May 1969.

16. Zohar Manna and Amir Pnueli, "Formalization of Properties of Recursively Defined Functions", Proc. ACM Symposium on Computing Theory, May 1969.

17. J. McCarthy and P. Hayes, "Some Philosophical Problems from the Standpoint of Artificial Intelligence", in D. Michie (ed), Machine Intelligence 4, American Elsevier, New York, 1969.

18. N. Nilsson, "A Mobile Automaton: An Application of Artificial Intelligence Techniques", Proc. International Conf. on Artificial Intelligence, Washington, D.C., 1969.

19. R. Paul, G. Falk, J. Feldman, "The Computer Representation of Simply Described Scenes", Proc. Illinois Graphics Conference, April 1969.

20. R. Schank and L. Tesler, "A Conceptual Parser for Natural Language", Proc. International Joint Conference On Artificial Intelligence, Washington, D.C., 1969.

21. G. Schroll, A. Duffield, C. Djerassi, B. Buchanan, G. Sutherland, E. Feigenbaum, and J. Lederberg, "Applications of Artificial Intelligence for Chemical Inference III. Aliphatic Ethers Diagnosed by Their Low Resolution Mass Spectra and NMR Data", J. American Chemical Society (in press).

APPENDIX B

THESES

Theses that have been published as Stanford Artificial Intelligence Memos are listed below. Several earned degrees at institutions other than Stanford. Abstracts of all A.I. Memos are given in Appendix D.

<u>Memo</u>	
AI-43	R. Reddy, "An Approach to Computer Speech Recognition by Direct Analysis of the Speech Wave", Ph.D. Thesis in Computer Science, Stanford University, September 1966.
AI-46	S. Persson, "Some Sequence Extrapolating Programs: A Study of Representation and Modeling in Inquiring Systems," Ph.D. Thesis in Computer Science, University of California, Berkeley, September 1966.
AI-47	B. Buchanan, "Logics of Scientific Discovery", Ph.D. Thesis in Philosophy, University of California, Berkeley, December 1966.
AI-44	J. Painter, "Semantic Correctness of a Compiler for an Algol-like Language," Ph.D. Thesis in Computer Science, Stanford University, March 1967.
AI-56	W. Wichman, "Use of Optical Feedback in the Computer Control of an Arm", Eng. Thesis in Electrical Engineering, Stanford University, August 1967.
AI-58	M. Callero, "An Adaptive Command and Control System Utilizing Heuristic Learning Processes", Ph.D. Theses in Operations Research, Stanford University, December 1967.
AI-63	D. Kaplan, "Regular Expressions and the Equivalence of Programs", Ph.D. Thesis in Computer Science, Stanford University, July 1968.
AI-65	B. Huberman, "A Program to Play Chess End Games", Ph.D. Thesis in Computer Science, Stanford University, August 1968.
AI-73	D. Pieper, "The Kinematics of Manipulators uncer Computer Control", Ph.D. Thesis in Mechanical Engineering, Stanford University, October 1968.
AI-74	D. Waterman, "Machine Learning of Heuristics", Ph.D. Thesis in Computer Science, Stanford University, December 1968.

THESES (cont.)

Memo

AI-83	R. Schank, "A Conceptual Dependency Representation for a Computer Oriented Semantics", Ph.D. Thesis in Linguistics, University of Texas, March 1969.
AI-85	P. Vicens, "Aspects of Speech Recognition by Computer", Ph.D. Thesis in Computer Science, Stanford University, March 1969.

APPENDIX C

FILM REPORTS

1. Art Eisenson and Gary Feldman, "Ellis D. Kropotechev and Zeus, his Marvelous Time-Sharing System", 16mm black and white with sound, runs about 15 minutes, March, 1967.

   <u>Abstract</u>: This film documents the advantages of time-sharing over standard batch processing. Through the good offices of the Zeus time-sharing system on the PDP-1 computer, our hero, Ellis, is saved from a fate worse than death. Recommended for mature audiences only.

2. Gary Feldman, "Butterfinger", 16mm color with sound, runs 8 minutes, March, 1968.

   <u>Abstract</u>: Describes the state of the hand-eye system at the Artificial Intelligence Project in the fall of 1967. The PDP-6 computer getting visual information from a television camera and controlling an electrical-mechanical arm solves simple tasks involving stacking blocks. The techniques of recognizing the blocks and their positions as well as controlling the arm are briefly presented.

3. Raj Reddy, Dave Espar and Art Eisenson, "Hear Here", 16mm color with sound, runs about 15 minutes, March, 1969.

   <u>Abstract</u>: Describes the state of the speech recognition project as of Spring, 1969. A discussion of the problems of speech recognition is followed by two real time demonstrations of the current system. The first shows the computer learning to recognize phrases and second show how the hand-eye system may be controlled by voice commands. Commands as complicated as "Pick up the small block in the lower lefthand corner," are recognized and the tasks required are carried out by the computer controlled arm.

4. Gary Feldman and Donald Peiper, "Avoid", 16mm silent,color, runs about 5 minutes, March, 1969.

   <u>Abstract</u>: Reports on a computer program written by D. Peiper for his Ph.D. thesis. The problem is to move the computer controlled electrical-mechanical arm through a space filled with one or more known obstacles. The programuses heuristics for finding a safe path; the film demonstrates the arm as it moves through various cluttered environments with fairly good success.

APPENDIX D ABSTRACTS OF ARTIFICIAL INTELLIGENCE PROJECT MEMOS

1963

1. J. McCarthy, Predicate Calculus with "undefined" as a Truth-Value, March
The use of predicate calculus in the matehmatical theory of computation and the problems involved in interpreting their values.

**2. J. McCarthy, Situations, Actions, and Causal Laws, July
A formal theory is given concerning situtations, causality and the possibility and effects of actions is given. The theory is intended to be used by the Advice Taker, a computer program that is to decide what to do by reasoning. Some simple examples are given of descriptions of situations and deductions that certain goals can be achieved.

3. F. Safier, "The Mikado" as an Advice Taker Problem, July
The situtation of the Second Act of "The Mikado" is analyzed from the point of view of Advice Taker formalism. This indicates defects still present in the language.

4. H. Enea, Clock Function for LISP 1.5, August
This paper describes a clock function for LISP 1.5.

5. H. Enea and D. Wooldridge, Algebraic Simplification, August
Herein described are proposed and effected changes and additions to Steve Russell's Mark IV Simplify.

*6. D. Wooldridge, Non-Printing Compiler, August
A short program which redefines parts of the LISP 1.5 compiler and suppresses compiler printout (at user's option) is described.

**7. J. McCarthy, Programs with Common Sense, September
Interesting work is being done in programming computers to solve probelms which require a high degree of intelligence in humans. However, certain elementary verbal reasoning processes so simple that they can be carried out by any non-feeble-minded human have yet to be simulated by machine programs.
This paper will discuss programs to manipulate in a suitable formal language (most likely a part of the predicate calculus)

* Out of print.
** Out of print. Reprinted as "Programs with Common Sense" in M. Minsky (Ed.), Semantic Information Processing, MIT Press, Cambridge, 1968.

common instrumental statements. The basic program will draw immediate conclusions from a list of premises. These conclusions will be either declarative or imperative sentences. When an imperative sentence is deduced the program takes a corresponding action. These actions may include printing sentences, moving sentences on lists, and reinitiating the basic deduction process on these lists.
Facilities will be provided for communication with humans in the system via manual intervention and display devices connected to the computer.

*8. J. McCarthy, Storage Conventions in LISP 2., September
Storage conventions and a basic set of functions for LISP 2 are proposed. Since the memo was written, a way of supplementing the features of this system with the unique storage of list structure using a hash rule for computing the address in a separate free storage area for lists has been found.

*9. C. M. Williams, Computing Estimates for the Number of Bisections of an N x N Checkerboard for N Even, December
This memo gives empirical justification for the assumption that the number of bisections of an N x N (N even) checkerboard is approximately given by the binomial coefficient $\binom{A}{\frac{A}{2}}$ where 2A is the length of the average bisecting cut.

10. S. R. Russell, Improvements in LISP Debugging, December
Experience with writing large LISP programs and helping students learning LISP suggests that spectacular improvements can be made in this area. These improvements are partly an elimination of sloppy coding in LISP 1.5, but mostly an elaboration of DEFINE, the push down list backtrace, and the current tracing facility. Experience suggests that these improvements would reduce the number of computer runs to debug a program a third to a half.

11. D. Wooldridge Jr., An Algebraic Simplify Program in LISP, December
A program which performs "obvious" (non-controversial)simplifying transformations on algebraic expressions (written in LISP prefix notation) is described. Cancellation of inverses and consolidation of sums and products are the basic accomplishments of the program; however, if the user desires to do so, he may request the program to perform special tasks, such as collect common factors from products in sums or expand products. Polynomials

* Out of print.

are handled by routines which take advantage of the special form by polynomials; in particular, division (not cancellation) is always done in terms of polynomials. The program (run on the IBM 7090) is slightly faster than a human; however, the computer does not need to check its work by repeating the simplification. Although the program is usable - no bugs are known to exist - it is by no means a finished project. A rewriting of the simplify system is anticipated; this will eliminate much of the existing redundancy and other inefficiency, as well as implement an identity-recognizing scheme.

1964

*12. G. Feldman, Documentation of the MacMahon Squares Problem, January
An exposition of the MacMahon Squares problem together with some "theoretical" results on the nature of its solutions and a short discussion of an ALGOL program which finds all solutions are contained herein.

13. D. Wooldridge, The New LISP System (LISP 1.55), February
The new LISP system is described. Although differing only slightly it is thought to be an improvement on the old system.

14. J. McCarthy, Computer Control of a Machine for Exploring Mars, January
Landing a 5000 pound package on Mars that would spend a year looking for life and making other measurements has been proposed. We believe that this machine should be a stored program computer with sense and motor organs and that the machine should be mobile. We discuss the following points. 1. Advantages of a computer controlled system. 2. What the computer should be like. 3. What we can feasibly program the machine to do given the present state of work on artificial intelligence. 4. A plan for carrying out research in computer controlled experiments that will make the Mars machine as effective as possible.

15. M. Finkelstein and F. Safier, Axiomatization and Implementation, June
An example of a typical Advice-Taker axiomatization of a situation is given, and the situation is programmed in LISP as an indication of how the Advice-Taker could be expected to react. The situation chosen is the play of a hand of bridge.

16. J. McCarthy, A Tough Nut for Proof Procedures, July
It is well known to be impossible to tile with dominoes a checkerboard with two opposite corners deleted. This fact is readily stated in the first order predicate calculus, but the usual proof which involves a parity and counting argument does not readily translate into predicate calculus. We conjecture that this problem will be very difficult for programmed proof procedures.

17. J. McCarthy, Formal Description of the Game of Pang-Ke, July
The game of Pang-Ke is formulated in a first-order-logic in order to provide grist for the Advice-Taker Mill. The memo does not explain all the terms used.

* Out of print.

1964 (cont.)

*18. J. Hext, An Expression input Routine for LISP, July
The expression input routine is a LISP function, Mathread [ ] with associated definitions, which reads in expressions such as (A+3 - F(X,Y,Z)). Its result is an equivalent S-expression. The syntax of allowable expressions is given, but (unlike ALGOL's) it does not define the precedence of the operators; nor does the program carry out any explicit syntax analysis. Instead, the program parses the expression according to a set of numerical precedence values, and reports if it finds any symbol out of context.

19. J. Hext, Programming Languages and Translation, August
A notation is suggested for defining the syntax of a language in abstract form, specifying only its semantic constituents. A simple language is presented in this form and its semantic definition given in terms of these constituents. Methods are then developed for translating this language, first into a LISP format and from there to machine code, and for proving that the translation is correct.

20. R. Reddy, Source Language Optimization of For-Loops, August
Program execution time can be reduced, by a considerable amount, by optimizing the 'For-loops' of Algol Programs. By judicious use of index-registers and by evaluating all the sub-expressions whose values are not altered within the 'For loop', such optimization can be achieved.
In this project we develop an algorithm to optimize Algol Programs in List-structure form and generate a new source language program, which contains the "desired contents in the index registers" as a part of the For-clause of the For-statement and additional statements for evaluating the same expressions outside the 'For-loop'. This optimization is performed only for the innermost 'For-loops'.
The program is written entirely in LISP. Arrays may have any number of subscripts. Further array declarations may have variable dimensions. (Dynamic allocation of storage.)
The program does not try to optimize arithmetic expressions. (This has already been extensively investigated.)

21. R. W. Mitchell, LISP 2 Specifications Proposal, August
Specifications for a LISP 2 system are proposed. The source language is basically ALGOL 60 extended to include list processing, input/output and language extension facilities. The system would be implemented with a source language translator and optimizer, the output of which could be processed by either an interpreter or a compiler. The implementation is

* Out of print.

specified for a single address computer with particular reference to an IBM 7090 where necessary.
Expected efficiency of the system for list processing is significantly greater than the LISP 1.5 interpreter and also somewhat better than the LISP 1.5 compiler. For execution of numeric algorithms the system should be comparable to many current "algebraic" compilers.
Some familiarity with LISP 1.5, ALGOL and the IBM 7090 is assumed.

22. R. Russell, Kalah - The Game and the Program, September
A description of Kalah and the Kalah program, including subroutine descriptions and operating instructions.

23. R. Russell, Improvements to the Kalah Program, September
Recent improvements to the Kalah program are listed, and a proposal for speeding up the program by a factor of three is discussed.

24. J. McCarthy, A Formal Description of a Subset of Algol, September
We describe Microalgol, a trivial subset of Algol, by means of an interpreter. The notions of abstract syntax and of "state of the computation" permit a compact description of both syntax and semantics. We advocate an extension of this technique as a general way of describing programming language.

25. R. Mansfield, A Formal System of Computation, September
We discuss a tentative axiomatization for a formal system of computation and within this system we prove certain propositions about the convergence of recursive definitions proposed by J. McCarthy.

26. R. Reddy, Experiments on Automatic Speech Recognition by a Digital Computer, October
Speech sounds have in the past been investigated with the aid of spectrographs, vo-coders and other analog devices. With the availability of digital computers with improved i-o devices such as Cathode Ray tubes and analog digital converters, it has recently become practicable to employ this powerful tool in the analysis of speech sounds.
Some papers have appeared in the recent literature reporting the use of computers in the determination of the fundamental frequency and for vowel recognition. This paper discusses the details and results of a preliminary investigation conducted at Stanford. It includes various aspects of speech sounds such as waveforms of vowels and consonants; determination of a fundamental of the wave; Fourier (spectral) analysis of the sound waves formant determination, simple vowel recognition

algorithm and synthesis of sounds. All were obtained by the use of a digital computer.

1965

27. J. McCarthy, A Proof-Checker for Predicate Calculus, March
A program that checks proofs in J. A. Robinson's formulation of predicate calculus has been programmed in LISP 1.5. The program is available in CTSS at Project MAC and is also available as a card deck. The program is used for class exercises at Stanford.

28. J. McCarthy, Problems in the Theory of Computation, March
The purpose of this paper is to identify and discuss a number of theoretical problems whose solutions seem feasible and likely to advance the practical art of computation. The problems that will be discussed include the following:
1. Semantics of programming languages. What do the strings of symbols representing computer programs, statements, declarations, labels, etc., denote? How can the semantics of programming languages be described formally?
2. Data spaces. What are the spaces of data on which computer programs act and how are they built up from simpler spaces?
3. How can time dependent and simultaneous processes be described?
4. Speed of computation. What can be said about how much computation is required to carry out certain processes?
5. Storage of information. How can information be stored so that items identical or similar to a given item can be retrieved?
6. Syntax directed computation. What is the appropriate domain for computations described by productions or other data format recognizers?
7. What are the appropriate formalisms for writing proofs that computer programs are equivalent?
8. In view of Gödel's theorem that tells us that any formal theory of computation must be incomplete, what is a reasonable formal system that will enable us to prove that programs terminate in practical cases?

29. C. M. Williams, Isolation of Important Features of a Multi-toned Picture, January
A roughly successful attempt is made to reduce a multi-toned picture to a two-toned (line drawing) representation capable of being recognized by a human being.

30. E. Feigenbaum and R. W. Watson, An Initial Problem Statement for a Machine Induction Research Project, April
A brief description is given of a research project presently getting under way. This project will study induction by machine, using organic chemistry as a task area. Topics for graduate student research related to the problem is listed.

1965 (cont.)

31. J. McCarthy, Plans for the Stanford Artificial Intelligence Project, April
The following is an excerpt from a proposal to ARPA and gives some of the project plans for the near future.

32. H. Ratchford, The 138 Analog Digital Converter, May
A discussion of the programming and hardware characteristics of the analog to digital converter on the PDP-1 is given; several sample programs are also presented.

33. B. Huberman, The Advice Taker and GPS, June
Using the formalism of the Newell-Shaw-Simon General Problem Solver to solve problems expressed in McCarthy's Advice Taker formalism is discussed. Some revisions of the formalism of can and cause described in AI Memo No. 2 are proposed.

34. P. Carah, A Television Camera Interface for the PDP-1, June
This paper is a discussion of several methods for the connection of a television camera to the PDP-1 computer. Three of these methods are discussed in detail and have in common that only a 36 bit portion of any horizontal scanning line may be read and this information is read directly into the working registers of the computer. The fourth involves a data channel to read information directly into the core memory of the computer, and is mentioned only in passing. The major concepts and some of the details of these methods are due to Marvin Minsky.

*35. F. Safier, Simple Simon, June
SIMPLE SIMON is a program which solves the problem of finding an object satisfying a predicate from a list of facts. It operates by backware chaining. The rules of procedure and heuristics are discussed and the structure of the program is outlined.

36. J. Painter, Utilization of a TV Camera on the PDP-1, September
A description of the programming required to utilize the TV camera connected to the PDP-1 and of the initial collection of programs.

37. K. Korsvold, An On Line Algebraic Simplification Program, November
We describe an on-line program for algebraic simplification. The program is written in LISP 1.5 for the Q-32 computer at System Development Corporation in Santa Monica, California. The program has in its entirety been written and debugged from a teletype station at Stanford University.

* Out of print.

1965 (cont.)

K. Korsvold, Appendix B, to A.I. 37
This appendix contains the program written in m-expressions. The four functions ADDK, TIMESKL, *GSD and *RFD are not included since they are written in LAP.

1966

38. D. Waterman, A Filter for a Machine Induction System, January
This report contains current ideas about the Machine Induction Research Project, and attempts to more clearly define some of the problems involved. In particular, the on-line data acquisition problem, the filter, and the inductive inference problem associated with the filter are discussed in detail.

39. K. Pingle, A Program to Find Objects in a Picture, January
A program is described which traces around objects in a picture, using the picture scanner attached to the PDP-1 computer, and fits curves to the edges.

40. J. McCarthy and J. Painter, Correctness of a Compiler for Arithmetic Expressions, April
This is a preprint of a paper given at the Symposium of Mathematical Aspects of Computer Science of the American Mathematical Society held April 7 and 8, 1966. It contains a proof of the correctness of a compiler for arithmetic expressions.

*41. P. Abrams and D. Rode, A Proposal for a Proof-Checker for Certain Axiomatic Systems, May
A proposed design for a proof-checker to operate on many axiomatic domains is presented. Included are descriptions of the organization and operation of the program to be written for the PDP-6.

42. K. Pingle, A Proposal for a Visual Input Routine, June
Some comments are made on the characteristics believed desirable in the next eye for the Stanford Artificial Intelligence Project and a proposal is given for a program to input scenes using the eye.

43. R. Reddy, An Approach to Computer Speech Recognition by Direct Analysis of the Speech Wave, September
A system for obtaining a phonemic transcription from a connected speech sample entered into the computer by a microphone and an analog-to-digital converter is described. A feature-extraction program divides the speech utterance into segments approximately corresponding to phonemes, determine pitch periods of those segments where pitch analysis is appropriate, and computes a list of parameters for each segment. A classification program assigns a phoneme-group label (vowel-like segment, fricative-like segment, etc.) to each segment, determines whether a segment should be classified as a phoneme or whether it represents a phoneme boundary between two phonemes, and then assigns a phoneme label to each segment that is not rejected as being a phoneme boundary. About 30 utterances of one to two seconds

* Out of print.

duration were analyzed using the above programs on an interconnected IBM 7090 - PDP1 system. Correct identification of many vowel and consonantal phonemes was achieved for a single speaker. The time for analysis of each utterance was about 40 times real time. The results were encouraging and point to a new direction in speech research.

44. J. Painter, Semantic Correctness of a Compiler for an Algol-like Language, Revised March, 1967
This is a semantic proof of the correctness of a compiler. The abstract syntax and semantic definition are given for the language Mickey, an extension of Micor-algol. The abstract syntax and semantics are given for a hypothetical one-register single-address computer with 14 operations. A compiler, using recursive descent, is defined. Formal definitions are also given for state vector, a and c functions, and correctness of a compiler. Using these definitions, the compiler is proven correct.

45. D. Kaplan, Some Completeness Results in the Mathematics Theory of Computation, October
A formal theory is described which incorporates the "assignment" function a(i, k, ξ) and the "contents" function c(i, ξ). The axioms of the theory are shown to comprise a complete and consistent set.

46. S. Persson, Some Sequence Extrapolating Programs: A Study of Representation and Modeling in Inquiring Systems, September.
The purpose of this thesis is to investigate the feasibility of designing mecahnized inquiring-systems for finding suitable representations of problems, i.e., to perform the "creative" task of finding analogies. Because at present a general solution to this problem does not seem to be within reach, the feasibility of mechanizing a particular representational inquirer is chosen as a reasonable first step towards an increased understanding of the general problem. It is indicated that by actually designing, programming and running a representational inquirer as a program for a digital computer, a severe test of its consistency and potential for future extensions can be performed.

*47. B.Buchanan, Logics of Scientific Discovery, December.

The concept of a logic of discovery is discussed from a philosophical point of view. Early chapters discuss the concept of discovery itself, some arguments which have been advanced against logics of discover, notably by N. R. Hanson, and

* Out of print. Available through University Microfilms, 300 N. Zeeb Road, P.O. Box 1346, Ann Arbor, Michigan 48106.

1966 (cont.)

S. E. Toulmin. While a logic of discovery is generally understood to be an algorithm for formulating hypotheses, other concepts have been suggested. Chapters V and VI explore two of these: (A) a set of criteria by which a hypotheses could be judged reasonable, and (B) a set of rational (but not necessarily effective) methods for formulating hypotheses.

1967

48. D. Kaplan, Correctness of a Compiler for Algol-like Programs, July
A compiling algorithm is given which maps a class of Algol-like programs into a class of machine language programs. The semantics, i.e., the effect of execution, of each class if specified, and recursion induction used to prove that program semantics is preserved under the mapping defined by the compiling algorithm.

49. G. Sutherland, DENDRAL - A Computer Program for Generating and Filtering Chemical Structures, February
A computer program has been written which can generate all the structural isomers of a chemical composition. The generated structures are inspected for forbidden substructures in order to eliminate structures which are chemically impossible from the output. In addition, the program contains heuristics for determining the most plausible structures, for utilizing supplementary data, and for interrogating the on-line user as to desired options and procedures. The program incorporates a memory so that past experiences are utilized in later work.

50. A. Hearn, Reduce Users' Manual, February
REDUCE is a program designed for general algebraic computations of interest to physicists and engineers. Its capabilities include:
1) expansion and ordering of rational functions of polynomials,
2) symbolic differentiation,
3) Substitutions in a wide variety of forms,
4) reduction of quotients of polynomials by cancellation of common factors,
5) calculation of symbolic determinants,
6) calculations of interest to high energy physicists including spin 1/2 and spin 1 algebra.

The program is written completely in the language LISP 1.5 and may therefore be run with little modification on any computer possessing a LISP 1.5 compiler or interpreter.

51. L. Earnest, Choosing an Eye for a Computer, April
In order for a computer to operate efficiently in an unstructured environment, it must have one or more manipulators (e.g., arms and hands) and a spatial sensor analogous to the human eye. Alternative sensor systems are compared here in their performance on certain simple tasks. Techniques for determining color, texture, and depth of surface elements are examined. Sensing elements considered include the photomultiplier, image dissector, image orthicon, vidicon, and SEC camera tube. Performance measures strongly favor a new (and undemonstrated) configuration that may be termed a laser jumping spot system.

52. A. L. Samuel, Some Studies in Machine Learning Using the Game of Checkers II - Recent Progress, June
A new signature table technique is described together with an improved book learning procedure which is thought to be much superior to the linear polynomial method described earlier. Full use is made of the so called "alpha-beta" pruning and several forms of forward pruning to restrict the spread of the move tree and to permit the program to look ahead to a much greater depth than it otherwise could do. While still unable to outplay checker masters, the program's playing ability has been greatly improved. Some of these newer techniques should be applicable to problems of economic importance.

53. B. Weiher, The PDP-6 Proof Checker, June
A description if given for the use of a proof checker for propositional calculus. An example of its use as well as the M and S expressions for the proof checker are also included.

54. J. Lederberg and E. A. Feigenbaum, Mechanization of Inductive Inference in Organic Chemistry, August
A computer program for formulating hypotheses in the area of organic chemistry is described from two standpoints: artificial intelligence and organic chemistry. The Dendral Algorithm for uniquely representing and ordering chemical structures defines the hypothesis-space; but heuristic search through the space is necessary because of its size. Both the algorithm and the heuristics are described explicitly but without reference to the LISP code in which these mechanisms are programmed. Within the program some use has been made of man-machine interaction, pattern recognition, learning, and tree-pruning heuristics as well as chemical heuristics which allow the program to focus its attention on a subproblem to rank the hypotheses in order of plausibility. The current performance of the program is illustrated with selected examples of actual output showing both its algorithmic and heuristic aspects. In addition some of the more important planned modifications are discussed.

55. J. Feldman, First Thoughts of Grammatical Inference, August.
A number of issues relating to the problem of inferring a grammar are discussed. A strategy for grammatical inference is presented and its weaknesses and possible improvements are discussed. This is a working paper and should not be reproduced, quoted or believed without the author's permission.

56. W. Wichman, Use of Optical Feedback in the Computer Control of an Arm, August.
This paper reports an experimental investigation of the application of visual feedback to a simple computer-controller block-stacking task. The system uses a vidicon camera to examine a table top containing two cubical blocks, generating a data structure which is analyzed to determine the position of one block. An electric arm picks up the block and removes it from the scene, then after the program locates the second block, places the first on top of the second. Finally, the alignment of the stack is improved by analysis of the relative position error as seen by the camera. Positions are determined throughout by perspective transformation of edges detected from a single viewpoint, using a support hypothesis to supply sufficient information on depth. The Appendices document a portion of the hardware used in the project.

57. A. C. Hearn, Reduce, A User-Oriented Interactive System for Algebraic Simplification, October
This paper describes in outline the structure and use of REDUCE, a program designed for large-scale algebraic computations of interest to applied mathematicians, physicists and engineers.
The capabilities of the system include:
1) expansion, ordering and reduction of rational functions of polynomials,
2) symbolic differentiation,
3) substitutions for variables and expressions appearing in other expressions,
4) simplification of symbolic determinants and matrix expressions,
5) tensor and non-commutative algebraic calculations of interest to high energy physicists.

In addition to the operations of addition, subtraction, multiplication, division, numerical exponentiation and differentiation, it is possible for the user to add new operators and define rules for their simplification. Derivations of these operators may also be defined.
The program is written complete in the language of LISP 1.5 and is organized so as to minimize the effort required in transferring from one LISP system to another.
Some particular problems which have arisen in using REDUCE in a time-sharing environment are also discussed.

58. M. D. Callero, An Adaptive Command and Control System Utilizing Heuristic Learning Processes, December
The objectives of the research reported here are to develop an automated decision process for real time allocation of defense missiles to attacking ballistic missiles in general war and to demonstrate the effectiveness of applying heuristic learning to seek optimality in the process. The approach is to model and simulate a missile defense environment and generate a

decision procedure featuring a self-modifying, heuristic decision function which improves its performance with experience. The goal of the decision process that chooses between the feasible allocations is to minimize the total effect of the attack, measured in cumulative loss of target value. The goal is pursued indirectly by considering the more general problem of maintaining a strong defense posture, the ability of the defense system to protect the targets from both current and future loss. Using a simulation and analysis, a set of calculable features are determined which effectively reflect the marginal deterioration of defense posture for each allocation in a time interval. A decision function, a linear polynomial of the features, is evaluated for each feasible allocation and the allocation having the smallest value is selected. A heuristic learning process is incorporated in the model to evaluate the performance of the decision process and adjust the decision function coefficients to encourage correct comparison of alternative allocations. Simulated attacks presenting typical defense situations were cycled against the decision procedure with the result that the decision function coefficients converged under the learning process and the decision process become increasingly effective.

1968

59. D. M. Kaplan, A Formal Theory Concerning the Equivalence of Algorithms, May
Axioms and rules of inference are given for the derivation of equivalence for algorithms. The theory is shown to be complete for certain subclasses of algorithms, and several applications of the theory are illustrated. This paper was originally presented at the Mathematical Theory of Computation Conference, IBM Yorktown Heights, November 27-30, 1967.

60. D. M. Kaplan, The Formal Theoretic Analysis of Strong Equivalence for Elemental Programs, June
The syntax and semantics is given for elemental programs, and the strong equivalence of these simple ALGOL-like flowcharts is shown to be undecidable. A formal theory is introduced for deriving statements of strong equivalence, and the completeness of this theory is obtained for various sub-cases. Several applications of the theory are discussed. Using a regular expression representation for elemental programs and an unorthodox semantics for these expressions, several strong equivalence detecting procedures are developed. This work was completed in essentially its present form March, 1968.

61. T. Ito, Notes of Theory of Computation and Pattern Recognition, May
This is a collection of some of the author's raw working notes during the period December 1965 - October 1967 besides the introduction. They have been privately or internally distributed for some time. Portions of this work have been accepted for publication; others are being developed for submission to journals. Some aspects and ideas have been referred to and used, sometimes without explicit references, and others are developed by other researchers and the author. Hence we have decided to publish this material as Computer Science Technical Report, although the author is planning to submit all of these works to some journals, adding several new results (not mentioned in this report), improving notations, definitions and style of presentation in some parts and reformulating completely in other parts. The author appreciates it very much of the researchers who use or refer to the results and ideas of this report communicate with him. The publication of this report was encouraged by Prof. George E. Forsythe and Prof. John McCarthy.

62. B. Buchanan and G. Sutherland, HEURISTIC DENDRAL: A Program for Generating Explanatory Hypotheses in Organic Chemistry, July
A computer program has been written which can formulate hypotheses from a given set of scientific data. The data consist

of the mass spectrum and the empirical formula of an organic chemical compound. The hypotheses which were produced describe molecular structures which are plausible explanations of the data. The hypotheses are generated systematically within the program's theory of chemical stability and within limiting constraints which are inferred from the data by heuristic rules. The program excludes hypotheses inconsistent with the data and lists its candidate explanatory hypotheses in order of decreasing plausibility. The computer program is heuristic in that it searches for plausible hypotheses in a small subset of the total hypothesis space according to heuristic rules learned from chemists.

63. D. M. Kaplan, Regular Expressions and the Equivalence of Programs, July
The strong equivalence of ALGOL-like programs is, in general, an undecidable property. Several mechanical procedures are discussed which nevertheless are useful in the detection of strong equivalence. These methods depend on a regular expression representation of programs. An unorthodox semantics for these expressions is introduced which appreciably adds to the ability to detect strong equivalence. Several other methods of extending this ability are also discussed.

64. Z. Manna, Formalization of Properties of Programs, July
Given a program, an algorithm will be described for constructing an expression, such that the program is valid (i.e., terminates and yields the right answer) if and only if the expression is inconsistent. Similar result for the equivalence problem of programs is given. These results suggest a new approach for proving the validity and the equivalence of programs.

65. B. Huberman, A Program to Play Chess End Games, August
A program to play chess end games is described. The model used in the program is very close to the model assumed in chess books. Embedded in the model are two predicates, better and worse, which contain the heuristics of play, different for each end game. The definitions of better and worse were obtained by programmer translation from the chess books.
The program model is shown to be a good one for chess and games by the success achieved for three end games. Also the model enables us to prove that the program can reach checkmate from any starting position. Insights about translation from book problem solving methods into computer program heuristics are discussed; they are obtained by comparing the chess book methods with the definitions of better and worse, and by considering the difficulty encountered by the programmer when doing the translation.

66. J. Feldman and P. Rovner, An Algol-Based Associative Language, August
A high-level programming language for large complex relational structures has been designed and implemented. The underlying relational data structure has been implemented using a hash-coding technique. The discussion includes a comparison with other work and examples of applications of the language. A version of this paper will appear in the communications of the ACM.

67. E. Feigenbaum, Artificial Intelligence: Themes in the Second Decade, August
In this survey of artificial Intelligence research, the substantive focus is heuristic programming, problem solving, and closely associated learning models. The focus in time is the period 1963-1968. Brief tours are made over a variety of topics: generality, integrated robots, game playing, theorem proving, semantic information processing, etc.
One program, which employs the heuristic search paradigm to generate explanatory hypotheses in the analysis of mass spectra of organic molecules, is described in some detail. The problem of representation for problem solving systems is discussed. Various centers of excellence in the artificial intelligence research area are mentioned. A bibliography of 76 references is given.

68. Z. Manna and A Pnueli, The Validity Problem of the 91-Function, August
Several methods for proving the weak and strong validity of algorithms are presented.
For proving the weak validity (i.e., correctness) we use satisfiability methods, while for proving the strong validity (i.e., termination and correctness) we use unsatisfiability methods.
Two types of algorithms are discussed: recursively defined functions and programs.
Among the methods we include known methods due to Floyd, Manna, and McCarthy. All the methods will be introduced quite informally by means of an example (the 91-function).

69. J. McCarthy, Project Technical Report, September.
Recent work of Stanford Artificial Intelligence Project is summarized in several areas:
Scientific Hypothesis Formation
Symbolic Computation
Hand-Eye Systems
Computer Recognition of Speech
Board Games
Other Projects

70. A. C. Hearn, The Problem of Substitution, December
One of the most significant features of programs designed for non-numeric calculation is that the size of expressions manipulated, and hence the amount of storage necessary, changes continually during the execution of the program. It is therefore usually not possible for the user to know ahead of time just how much output his program will produce, or whether the calculation will in fact fail because of lack of available computer memory. The key to keeping both the size of intermediate expressions and output under control often lies in the manner in which substitutions for variables and expressions declared by the programmer are implemented by the system. In this paper various methods which have been developed to perform these substitutions in the author's own system REDUCE are discussed. A brief description of the REDUCE system is also given.

71. P. Vicens, Preprocessing for Speech Analysis, October
This paper describes a procedure, and its hardware implementation, for the extraction of significant parameters of speech. The process involves division of the speech spectrum into convenient frequency bands, and calculation of amplitude and zero-crossing parameters in each of these bands every 10 ms. In the software implementation, a smooth function divides the speech spectrum into two frequency bands (above and below 1000 Hz). In the hardware implementation, the spectrum is divided into three bands using bandpass filters (150-900 Hz, 900-2200 Hz, 2200-5000 Hz). Details of the design and implementation of the hardware device are given.

72. D. L. Pieper, The Kinematics of Manupulators Under Computer Control, October
The kinematics of manipulators is studied. A model is presented which allows for the systematic description of new and existing manipulators.
Six degree-of-freedom manipulators are studied. Several solutions to the problem of finding the manipulator configuration leading to a specified position and orientation are presented. Numerical as well as explicit solutions are given. The problem of positioning a multi-link digital arm is also discussed.
Given the solution to the position problem, as a set of heuristics is developed for moving a six degree-of-freedom manipulator from an initial position to a final position through a space containing obstacles. This results in a computer program shown to be able to direct a manipulator around obstacles.

<u>1968 (cont.)</u>

73. John McCarthy, <u>Some Pilosophical Problems from the Standpoint of Artificial Intelligence</u>, November

A computer program capable of acting intelligently in the world must have a general representation of the world in terms of which its inputs are interpreted. Designing such a program requires commitments about what knowledge is and how it is obtained. Thus some of the major traditional problems of philosophy arise in artificial intelligence.

More specifically, we want a computer program that decides what to do by inferring in a formal language that a certain strategy will achieve its assigned goal. This requires formalizing concepts of causality, ability, and knowledge. Such formalisms are also considered in philosophical logic.

The first part of the paper begins with a philosophical point of view that seems to arise naturally once we take seriously the idea of actually making an intelligent machine. We go on to the notions of metaphysically and epistemologically adequate representations of the world and then to an explanation of <u>can</u>, <u>causes</u>, and <u>knows</u>, in terms of a representation of the world by a system of interacting automata. A proposed resolution of the problem of freewill in a deterministic universe and of counterfactual conditional setences is presented.

The second part is mainly concerned with formalisms within which it can be proved that a strategy will achieve a goal. Concepts of situation, fluent, future operator, action, strategy, result of a strategy and knowledge are formalized. A method is given of constructing a sentence of first order logic which will be true in all models of certain axioms if and only if a certain strategy will achieve a certain goal.

The formalism of this paper represents an advance over (McCarthy 1963) and (Green 1968) in that it permits proof of the correctness of strategies that contain loops and strategies that involve the acquisition of knowledge, and it is also somewhat more concise.

The third part discusses open problems in extending the formalism of Part II.

The fourth part is a review of work in philosophical logic in relation to problems of artificial intelligence and discussion of previous efforts to program "general intelligence" from the point of view of this paper. This paper is based on a talk given to the 4th Machine Intelligence Workshop held at Edinburgh, August 12-21, 1968, and is a preprint of a paper to be published in 'Machine Intelligence 4' (Edinburgh University Press, 1969).

1968 (cont'd.)

*74. D. Waterman, Machine Learning of Heuristics,
The research reported here is concerned with devising machine-learning techniques which can be applied to the problem of automating the learning of heuristics.

75. R.C. Schank, A Notion of Linguistic Concept: A Prelude to Mechanical Translation
The conceptual dependency framework has been used as an automatic parser for natural language. Since the parser gives as output a conceptual network capable of expressing meaning in language-free terms it is possible to regard this as an interlingua. If an interlingua is actually available how might this interlingua be used in translation? The primary problem that one encounters is the definition of just what these concepts in the network are. A concept is defined as an abstraction in terms of percepts and the frequency of connection of other concepts. This definition is used to facilitate the understanding of some of the problems in paraphrasing and translation. The motivation for this abstract definition of linguistic concept is discussed in the context of its proposed use.
DESCRIPTORS: Computational Linguistics, Concepts Research, Computer Understanding.

76. R.C. Schank, A Conceptual Parser for Natural Language,
This paper describes an operable automatic parser for natural language. The parser is not concerned with producing the syntactic structure of an input sentence. Instead, it is a conceptual parser, concerned with determining the underlying meaning of the input. The output of the parser is a network of concepts explicating the conceptual relationships in a piece of discourse. The structure of this network is language-free; thus, sentences in different languages or paraphrases within the same language will parse into the same network. The theory behind this representation is outlined in this paper and the parsing algorithm is explained in some detail.
DESCRIPTORS: Computational Linguistics, Concepts, Linguistic Research, Computer Understanding.

*Out of print.

1969

77. J. D. Becker, The Modeling of Simple Analogic and Inductive Processes in a Semantic Memory System, January

In this paper we present a general data structure for a semantic memory, which is distinguished in that a notion of consequence (temporal, causal, logical, or behavioral, depending on interpretation) is a primitive of the data representation. The same item of data may at one time serve as a logical implication, and at another time as a "pattern/action" rule for behavior.

We give a definition of "analogy" between items of semantic information. Using the notions of consequence and analogy, we construct an inductive process in which general laws are formulated and verified on the basis of observations of individual cases. We illustrate in detail the atainment of the rule "Firemen wear red suspenders" by this process.

Finally, we discuss the relationship between analogy and induction, and their use in modeling aspects of "perception" and "understanding".

78. D. R. Reddy, On the Use of Environmental, Syntactic, and Probalistic Constraints in Vision and Speech, January

In this paper we consider both vision and speech in the hope that a unified treatment, illustrating the similarities, would lead to a better appreciation of the problems, and possibly programs which use the same superstructure. We postulate a general perceptual system and illustrate how various existing systems either avoid or ignore some of the difficult problems that must be considered by a general perceptual system. The purpose of this paper is to point out some of the unsolved problems, and to suggest some heuristics that reflect environmental, syntactic, and probabilistic constraints useful in visual and speech perception by machine. To make effective use of these heuristics, a program must provide for

1. An external representation of heuristics for ease of man-machine communication
2. An internal representation of heuristics for effective use by machine
3. A mechanism for the selection of appropriate heuristics for use in a given situation.

Machine perception of vision and speech, thus, provides a problem domain for testing the adequacy of the models of representation (McCarthy and Hayes), planning and heuristic selection (Minsky, Newell and Simon), and generalization learning (Samuel); a domain in which (perceptual) tasks are performed by people easily and without effort.

79. D. R. Reddy and R. B. Neely, Contextual Analysis of Phonemes of English, January

It is now well known that the acoustic characteristics of a

Phoneme depend on both the preceding and following phonemes. This paper provides some needed contextual and probabilistic data about trigram phonemic sequences of spoken English. Since there are approximately 40^3 such sequences, one must discover and study only the more commonly occurring sequences. To this purpose, three types of tables are presented, viz.,
a. Commonly occurring trigram sequences of the form $/\alpha\beta\gamma/$ for every phoneme $/\beta/$.
b. Commonly occurring sequences $/\alpha\beta\gamma/$ for every pair of phonemes $/\alpha/$ and $/\gamma/$.
c. Commonly occurring word boundary sequences of the form $/-\alpha\beta/$ and $/\alpha\beta-/$ where $/-/$ represents the silence phoneme.
Entries of the above tables contain examples of usage and probabilities of occurrence for each such sequence.

80. Georgia Sutherland, Heuristic Dendral: A Family of LISP Programs, March
The Heuristic Dendral program for generating explanatory hypotheses in organic chemistry is described as an application of the programming language LISP. The description emphasizes the non-chemical aspects of the program, particularly the "topologist" which generates all tree graphs of a collection of nodes.

81. David Luckham, Refinement Theorems in Resolution Theory, March
The paper discusses some basic refinements of the Resolution Principle which are intended to improve the speed and efficiency of theorem-proving programs based on this rule of inference. It is proved that two of the refinements preserve the logical completeness of the proof procedure when used separately, but not when used in conjunction. The results of some preliminary experiments with the refinements are given. Presented at the IRIA symposium on Automatic Deduction, Versailles, France, December 16-21, 1968.

82. Zohar Manna and Amir Pneuli, Formalization of Properties of Recursively Defined Functions, March
This paper is concerned with the relationship between the convergence, correctness and equivalence of recursively defined functions and the satisfiability (or unsatisfiability) of certain first-order formulas.

83. Roger C. Schank, A Conceptual Dependency Representation for a Computer-Oriented Semantics, March
Machines that may be said to function intelligently must be able to understand questions posed in natural language. Since natural language may be assumed to have an underlying conceptual structure, it is desirable to have the machine structure its own experience, both linguistic and nonlinguistic, in a manner concomitant with the human method for doing

so. Some previous attempts at organizing the machine's data base conceptually are discussed. A conceptually-oriented dependency grammar is posited as an interlingua that may be used as an abstract representation of the underlying conceptual structure. The conceptual dependencies are utilized as the highest level in a stratified system that incorporates language-specific realization rules to map from concepts and their relations, into sentences. In order to generate coherent sentences, a conceptual semantics is posited that limits possible conceptual dependencies to statements about the system's knowledge of the real world. This is done by the creation of semantic files that serve to spell out the defining characteristics of a given concept and enumerate the possibilities for relations with other concepts within the range of conceptual experience. The semantic files are created, in part, from a hierarchical organization of semantic categories. The semantic category is part of the definition of a concept and the information at the nodes dominating the semantic category in the hierarchical tree may be used to fill in the semantic file. It is possible to reverse the realization rules to operate on sentences and produce a conceptual parse. All potential parses are checked with the conceptual semantics in order to eliminate semantic and syntactic ambiguities. The system has been programmed; coherent sentences have been generated and the parser is operable. The entire system is posited as a viable linguistic theory

84. David Canfield Smith, MLISP Users' Manual, January
MLISP is a LISP pre-processor designed to facilitate the writing, use, and understanding of LISP programs. This is accomplished through parentheses reduction, comments, introduction of a more visual flow of control with block structure and mnemonic key words, and language redundancy. In addition, some "meta-constructs" are introduced to increase the power of the language.

85. Pierre Vicens, Aspects of Speech Recognition by Computer, April
This thesis describes techniques and methodology which are useful in achieving close to real-time recognition of speech by computer. To analyze connected speech utterances, any speech recognition system must perform the following processes: preprocessing, segmentation, segment classification, recognition of words, recognition of sentences. We present implemented solutions to each of these problems which achieved accurate recognition in all the trial cases.

1969 (cont'd)

86. Patrick J. Hayes, A Machine-Oriented Formulation of the Extended Functional Calculus, April

The Extended Functional Calculus (EFC), a three-valued predicate calculus intended as a language in which to reason about the results of computations, is described in some detail. A formal semantics is given. A machine-oriented (axiomless) inference system for EFC is then described and its completeness relative to the semantics is proved by the method of Semantic Trees. Finally some remarks are made on efficiency.

87. John McCarthy, and A.I. Project Staff, Project Technical Report, June

Plans and accomplishments of the Stanford Artificial Intellligence Project are reviewed in several areas including:

theory (epistemology and mathematical theory of computation),
visual perception and control (Hand-eye and Cart),
speech recognition by computer,
heuristics in machine learning and automatic deduction,
models of cognitive processes (Heuristic DENDRAL, Language Research, and Higher Mental Functions).

This is an excerpt of a proposal to ARPA.

88. Roger C. Schank, Linguistics from a Conceptual Viewpoint (Aspects of Aspects of a Theory of Syntax), April

Some of the assertions made by Chomsky in Aspects of the Theory of Syntax are considered. In particular, the notion of a 'competence' model in linguistics is criticized. Formal postulates for a conceptually-based linguistic theory are presented.

APPENDIX E

STANFORD ARTIFICIAL INTELLIGENCE LABORATORY

OPERATING NOTES

The SAILON series describes the operation of computer programs and equipment used in the Stanford Artificial Intelligence Laboratory.

1) (Superseded by SAILON 28.2, Appendix I.)

2.1) W. Weiher, "Calcomp Plot Routines", revised September 1968.
Describes use of basic plot routines from either FORTRAN IV or MACRO.

3.1) B. Baumgart, "How to Do It and Summaries of Things", revised March 1969. An introductory summary of system features.

4) (Superseded by SAILON 28.2.)

5) W. Weiher, "Preliminary Description of EDIT 2", January 1967.
Describes a Dectape-oriented teletype editor (obsolescent).

6) J. Sauter, "Stanford PDP-6 Time-Sharing System Documentation", January 1967.
Describes some differences between the Stanford Time-Sharing Monitor and the DEC system, including "Spacewar Mode" for real time programs.

7) (Obsolete)

8) S. Russell, "Recent Additions to FORTRAN Library", March 1967.

9) P. Petit, "Electronic Clock", March 1967.
Electronic clock attached to the system gives time in micro-seconds, seconds, minutes, hours, day, month, and year.

10) (Obsolete)

11) P. Petit, "A Recent Change to the Stanford PDP-6 Hardware," March 1967.
The PDP-6 has been changed so that user programs can do their own I/O to devices numbered 700 and above.

12) through 20) (Obsolete)

21) A. Grayson, "The A-D Converter", June 1967.

21 Addendum 1) E. Panofsky, "A/D Converter Multiplexer Patch Panel and Channel Assignments as of 1/9/69", January 1969.

22) (Obsolete)

23) W. Weiher, "Recent Changes to Various System Programs", July 1967.
Describes some modifications to TECO, MACROX, DDT, and LOADER.

24) S. Russell, "PDP-6 I/O Device Number Summary", August 1967.

25) S. Russell, "The Miscellaneous Outputs," August 1967.
Gives bit assignments for output to hydraulic arm and TV camera positioning.

26.1) P. Petit, "FAIL", revised October 1968.
Describes one-pass assembler that is about five times as fast as MACRO and has a more powerful macro processor.

27) P. Land, "PIP", August 1967.
A general file handler (DEC manual is better.).

28.2) L. Quam, "Stanford LISP 1.6 Manual", revised December 1968.
Describes the LISP interpreter and compiler, the editor ALVINE, and other aspects of this venerated list processing system.

29) W. Weiher, "Preliminary Description of the Display Processor", August 1967.
III display system from the programmer's viewpoint.

30) R. Paul, "Operation of the Image Dissector", November 1967.
Theory and operation of the III image dissector camera.

31) J. Sauter, "Disc Diagnostic", October 1967.
A program to test the Librascope Disk and its interface.

32) S. Russell and R. Gruen, "Aid for On-line Computation", October 1967.
Interactive calculator derived from JOSS (DEC manual is better.).

33) K. Pingle, "A List-Processing System for Picture Processing", October 1967.
A set of macros for creating complex data structures in the hand-eye system.

34) DEC Library, "PDP-6 Time Sharing TECO", October 1967.
Interactive string processing system for text editing.

35) K. Pingle, "Hand-Eye Library File", June 1968.

36) G. Feldman, "Fourier Transform Subroutine", June 1968.
FORTRAN subroutine performs one-dimensional Fast Fourier Transform.

37) S. Russell and L. Earnest, "A.I. Laboratory Users Guide", June 1968.
Orientation and administrative procedures.

37 Supplement 1) J. McCarthy, "A.I. Laboratory Users Guide", June 1968.
Hard-line administration.

38) P. Vicens, "New Speech Hardware", August 1968.
Preprocessor for input to speech recognition systems.

39) J. Sauter and D. Swinehart, "SAVE", August 1968.
Program for saving and restoring a single user's disk files on magnetic tape.

40) (Obsolete)

41) L. Quam, "SMILE at LISP", September 1968.
A package of useful LISP functions.

42) G. Falk, "Vidicon Noise Measurements", September 1968.
Measurements of spatial and temporal noise on Cohu vidicon camera connected to the computer.

43) A. Moorer, "DAEMON - Disk Dump and Restore", September 1968.
Puts all or selected files on magnetic tape.

44) A. Moorer, "FCROX - MACROX to Fail Converter", September 1968.
Converts MACROX programs to FAIL format, with a few annotated exceptions.

45) A. Hearn, "REDUCE Implementation Guide", October 1968.
Describes the procedure for assembling REDUCE (a symbolic computation system) in any LISP system.

46) W. Weiher, "Loader Input Format", October 1968.

47 and 47 Supplement 1) J. Sauter and J. Singer, "Known Programming Differences Between the PDP-6 and PDP-10", November 1968.

48) D. Swinehart, "GOGOL III", December 1968.
An algebraic language remeniscent of ALGOL. Includes variable length strings.

49) A. Hearn, "Service Routines for Standard LISP Users", February 1969.

50) W. Weiher, "STOPGAP", February 1969.
A disk-oriented teletype editor.

51) W. Weiher and B. Baumgart, "RPG, Rapid Program Generation", March 1969.
A set of Time Sharing Monitor commands that simplifies the running of text editors, assemblers, compilers, and the loader.

52) A. Moorer, "System Bootstrapper's Manual", February 1969.
How to bring back the system from various states of disarray.