Stanford Heuristic Programming Project
Memo HPP-79-23

Computer Science Department
Report No. STAN-CS-79-757

August 1979

LEVEL

Applications-oriented AI Research: Medicine

by

Victor B. Ciesielski, James S. Bennett, and Paul R. Cohen

a section of the

Handbook of Artificial Intelligence

edited by

Avron Barr and Edward A. Feigenbaum

COMPUTER SCIENCE DEPARTMENT
School of Humanities and Sciences
STANFORD UNIVERSITY

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: HPP-79-23 (STAN-CS-79-757, HPP-79-23)	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER: Technical Rept.
4. TITLE (and Subtitle): Applications-oriented AI Research: Medicine	5. TYPE OF REPORT & PERIOD COVERED: technical, August 1979
	6. PERFORMING ORG. REPORT NUMBER: HPP-79-23 (STAN-CS-79-757)
7. AUTHOR(s): Victor B. Ciesielski, James S. Bennett, and Paul R. Cohen, (edited by Avron Barr & Edward A. Feigenbaum)	8. CONTRACT OR GRANT NUMBER(s): MDA 903-77-C-0322, ARPA Order-3423
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Department of Computer Science, Stanford University, Stanford, California 94305 USA	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: Defense Advanced Research Projects Agency, Information Processing Techniques Office, 1400 Wilson Ave., Arlington, VA 22209	12. REPORT DATE: August 1979
	13. NUMBER OF PAGES: 53
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Mr. Philip Surra, Resident Representative, Office of Naval Research, Durand 165, Stanford University	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

DISTRIBUTION STATEMENT A
Approved for public release; Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

62

18. SUPPLEMENTARY NOTES

RR00785

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

RR0078506

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

(see reverse side)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

094120

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Those of us involved in the creation of the Handbook of Artificial Intelligence, both writers and editors, have attempted to make the concepts, methods, tools, and main results of artificial intelligence research accessible to a broad scientific and engineering audience. Currently, AI work is familiar mainly to its practicing specialists and other interested computer scientists. Yet the field is of growing interdisciplinary interest and practical importance. With this book we are trying to build bridges that are easily crossed by engineers, scientists in other fields, and our own computer science colleagues.

In the Handbook we intend to cover the breadth and depth of AI, presenting general overviews of the scientific issues, as well as detailed discussions of particular techniques and important AI systems. Throughout we have tried to keep in mind the reader who is not a specialist in AI.

As the cost of computation continues to fall, new areas of computer applications become potentially viable. For many of these areas, there do not exist mathematical "cores" to structure calculational use of the computer. Such areas will inevitably be served by symbolic models and symbolic inference techniques. Yet those who understand symbolic computation have been speaking largely to themselves for twenty years. We feel that it is urgent for AI to "go public" in the manner intended by the Handbook.

Several other writers have recognized a need for more widespread knowledge of AI and have attempted to help fill the vacuum. Lay reviews, in particular Margaret Boden's **Artificial Intelligence and Natural Man**, have tried to explain what is important and interesting about AI, and how research in AI progresses through our programs. In addition, there are a few textbooks that attempt to present a more detailed view of selected areas of AI, for the serious student of computer science. But no textbook can hope to describe all of the sub-areas, to present brief explanations of the important ideas and techniques, and to review the forty or fifty most important AI systems.

The Handbook contains several different types of articles. Key AI ideas and techniques are described in core articles (e.g., basic concepts in heuristic search, semantic nets). Important individual AI programs (e.g., SHRDLU) are described in separate articles that indicate, among other things, the designer's goal, the techniques employed, and the reasons why the program is important. Overview articles discuss the problems and approaches in each major area. The overview articles should be particularly useful to those who seek a summary of the underlying issues that motivate AI research.

Eventually the Handbook will contain approximately two hundred articles. We hope that the appearance of this material will stimulate interaction and cooperation with other AI research sites. We look forward to being advised of errors of omission and commission. For a field as fast moving as AI, it is important that its practitioners alert us to important developments, so that future editions will reflect this new material. We intend that the **Handbook of Artificial Intelligence** be a living and changing reference work.

The Handbook represents the work of many graduate students at Stanford as well as students and AI professionals at other institutions, including Rutgers University, SRI International, Xerox Palo Alto Research Center, MIT, and the RAND Corporation. This report on research toward applying AI techniques in medical systems was originally drafted at Rutgers University by Victor Ciesielski and his colleagues there. James Bennet and Paul Cohen at Stanford continued work on the material. Others who contributed to or commented on earlier versions of this section include Saul Amarel, Donald Biesel, Bruce Buchanan, Randall Davis, Casimir Kulikowsky, Donald Smith, and William Swartout.

Applications-oriented AI Research: Medicine

by

Victor B. Ciesielski, James S. Bennett, and Paul R. Cohen

a section of the

Handbook of Artificial Intelligence

edited by

Avron Barr and Edward A. Feigenbaum

This research was supported by both the Defense Advanced Research Projects Agency (ARPA Order No. 3423, Contract No. MDA 903-77-C-0322) and the National Institutes of Health (Contract No. NIH RR-00785-06). Victor Ciesielski's work was supported by the National Instututes of Health at the Laboratory for Computer Science at Rutgers University (Contract No. NIH RR-643). The views and conclusions of this document should not be interpreted as necessarily representing the official policies, either express or implied, of the Defense Advanced Research Projects Agency, the National Institutes of Health, or the United States Government.

Applications-oriented AI Research: Medicine

Table of Contents

Foreword

Those of us involved in the creation of the **Handbook of Artificial Intelligence**, both writers and editors, have attempted to make the concepts, methods, tools, and main results of artificial intelligence research accessible to a broad scientific and engineering audience. Currently, AI work is familiar mainly to its practicing specialists and other interested computer scientists. Yet the field is of growing interdisciplinary interest and practical importance. With this book we are trying to build bridges that are easily crossed by engineers, scientists in other fields, and our own computer science colleagues.

In the Handbook we intend to cover the breadth and depth of AI, presenting general overviews of the scientific issues, as well as detailed discussions of particular techniques and important AI systems. Throughout we have tried to keep in mind the reader who is not a specialist in AI.

As the cost of computation continues to fall, new areas of computer applications become potentially viable. For many of these areas, there do not exist mathematical "cores" to structure calculational use of the computer. Such areas will inevitably be served by symbolic models and symbolic inference techniques. Yet those who understand symbolic computation have been speaking largely to themselves for twenty years. We feel that it is urgent for AI to "go public" in the manner intended by the Handbook.

Several other writers have recognized a need for more widespread knowledge of AI and have attempted to help fill the vacuum. Lay reviews, in particular Margaret Boden's **Artificial Intelligence and Natural Man**, have tried to explain what is important and interesting about AI, and how research in AI progresses through our programs. In addition, there are a few textbooks that attempt to present a more detailed view of selected areas of AI, for the serious student of computer science. But no textbook can hope to describe all of the sub-areas, to present brief explanations of the important ideas and techniques, and to review the forty or fifty most important AI systems.

The Handbook contains several different types of articles. Key AI ideas and techniques are described in core articles (e.g., basic concepts in heuristic search, semantic nets). Important individual AI programs (e.g., SHRDLU) are described in separate articles that indicate, among other things, the designer's goal, the techniques employed, and the reasons why the program is important. Overview articles discuss the problems and approaches in each major area. The overview articles should be particularly useful to those who seek a summary of the underlying issues that motivate AI research.

Eventually the Handbook will contain approximately two hundred articles. We hope that the appearance of this material will stimulate interaction and cooperation with other AI research sites. We look forward to being advised of errors of omission and commission. For a field as fast moving as AI, it is important that its practitioners alert us to important developments, so that future editions will reflect this new material. We intend that the **Handbook of Artificial Intelligence** be a living and changing reference work.

The Handbook represents the work of many graduate students at Stanford as well as students and AI professionals at other institutions, including Rutgers University, SRI International, Xerox Palo Alto Research Center, MIT, and the RAND Corporation. This report on research toward applying AI techniques in medical systems was originally drafted at Rutgers University by Victor Ciesielski and his colleagues there. James Bennet and Paul Cohen at Stanford continued work on the material. Others who contributed to or commented on earlier versions of this section include Saul Amarel, Donald Biesel, Bruce Buchanan, Randall Davis, Casimir Kulikowsky, Donald Smith, and William Swartout.

Avron Barr
Edward Feigenbaum

Stanford University
July, 1979

Handbook of Artificial Intelligence

Topic Outline

Volumes I and II

Introduction

The Handbook of Artificial Intelligence
Overview of AI Research
History of AI
An Introduction to the AI Literature

Search

Overview
Problem Representation
Search Methods for State Spaces, AND/OR Graphs, and Game Trees
Six Important Search Programs

Representation of Knowledge

Issues and Problems in Representation Theory
Survey of Representation Techniques
Seven Important Representation Schemes

AI Programming Languages

Historical Overview of AI Programming Languages
Comparison of Data Structures and Control Mechanisms in AI Languages
LISP

Natural Language Understanding

Overview - History and Issues
Machine Translation
Grammars
Parsing Techniques
Text Generation Systems
The Early NL Systems
Six Important Natural Language Processing Systems

Speech Understanding Systems

Overview - History and Design Issues
Seven Major Speech Understanding Projects

Applications-oriented AI Research -- Part 1

Overview
TEIRESIAS - Issues in Expert Systems Design
Research on AI Applications in Mathematics (MACSYMA and AM)
Miscellaneous Applications Research

Applications-oriented AI Research -- Part 2: Medicine

Overview of Medical Applications Research
Six Important Medical Systems

Applications-oriented AI Research -- Part 3: Chemistry

Overview of Applications in Chemistry
Applications in Chemical Analysis
The DENDRAL Programs
CRYSALIS
Applications in Organic Synthesis

Applications-oriented AI Research -- Part 4: Education

Historical Overview of AI Research in Educational Applications
Issues in ICAI Systems Design
Seven Important ICAI Systems

Automatic Programming

Overview
Techniques for Program Specification
Approaches to AP
Eight Important AP Systems

The following sections of the Handbook are still in preparation and will appear in the third volume:

Theorem Proving
Vision
Robotics
Information Processing Psychology
Learning and Inductive Inference
Planning and Related Problem-solving Techniques

A. Overview

There are two main areas where AI techniques are being applied in medical systems Although the application of pattern recognition and scene analysis techniques to the interpretation of x-ray and ultrasonic images is an increasingly important diagnostic tool, this report will focus on another area, the construction of consultation programs as an aide in medical decision making.

The motivation for the development of expert computer-based medical consultation systems is twofold: First, there are obvious benefits to society from providing reliable and thorough diagnostic services--perhaps even at a reduced cost. It has been observed (Ledley & Lusted, 1959) that most of the errors made by clinicians are errors of omission, that is, in trying to identify the disease that a patient is suffering from, the physician does not consider all of the possibilities, thereby missing the correct diagnosis. A computer program could be designed to exhaustively consider all of the diseases in its domain. Furthermore, there are some tasks that computers can perform more rapidly and accurately, such as calculating doses of medicines, particularly in cases where dosage is critical and many factors must be taken into account in the calculation (as in digitalis therapy, see Article C5). There are also some tasks that physicians are notoriously poor at performing and that are routine enough for the computer to do, such as the prescription of anti-microbial therapy.

The second motivation for development of these systems stems from current interests in computer science. Clinical medicine has been a very fertile area for the study of cognitive processes ever since the diagnostic process has been studied extensively (Jacquez, 1963). There is a highly developed medical taxonomy; a large, relatively well-organized knowledge base; and a number of human experts in the domain whose performance is significantly better on hard problems that that of the average practitioner (i.e., there is an identifiable expertise). Furthermore, the type of problem solving that occurs in the domain is repetitive. These attributes reflect some of the prerequisites for applications of a developing sub-field of AI known as *knowledge engineering*--taking AI beyond the stage of "toy" problems to confront large, real-world problems (Feigenbaum, 1977).

Computer-based consultation brings with it many formidable social, psychological, and ethical problems that must be addressed by the system builders. These problems include: validating the systems, exporting them to hospitals and clinics, getting physicians and patients to accept them, and deciding the responsibility for decisions made by these systems.

In the following sections, aspects of the diagnostic process and medical decision making will be discussed, as well as a number of AI issues related to the representation and manipulation of medical knowledge.

Medical Decision Making

There are three principal parts of medical decision making: data gathering, diagnosis, and treatment recommendation. Data gathering is concerned with obtaining the patient history and clinical and laboratory data. The clinical data consist of symptoms, which are the subjective sensations reported by the patient--such as headache, chest pain, etc.--and

signs, which are objective and observable by the physician (Feinstein, 1967). **Manifestation** refers to any sign, symptom, or finding. Laboratory results generally are referred to as **findings**. Diagnosis is the process of using this data to determine the illness. The three aspects are not independent; disease hypotheses are used to direct further information gathering, while treatment recommendation depends on the diagnosis and generally requires more information gathering. Often, the decision to do a test includes a physician's estimate of the cost, both in terms of money and danger to the patient, which is weighed against the value of the information gained. Gathering information, diagnosing the disease, and deciding on a treatment regimen constitute a **consultation**. Figure 1 illustrates this process in relation to the course of the disease.

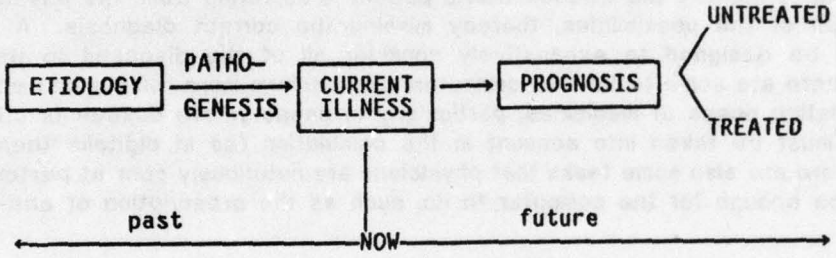


Figure 1. Consultation process depicting current state of medical knowledge.

This characterization of a consultation highlights the current state of medical knowledge. Etiology refers to the ultimate causes of the disease; pathogenesis refers to the way in which the disease developed from its causes. A consultation proceeds by determining the etiology. A treatment is then formulated for the identified diseases and their causes. Often, however, the medical knowledge is incomplete and it is not possible to determine the causes of a disease. In these cases, treatments must be based only on the knowledge of the symptoms or characteristics of the diseases. Some diseases are very well understood and knowledge about them is based on various kinds of models and specific mechanisms. Other diseases are not very well understood and knowledge about them is only associational; for example, treatment is prescribed on the basis of symptoms of closely associated diseases for which treatments are known.

During a consultation the physician performs at least two mental processes: reasoning and judgment (Ledley & Lusted, 1959). Reasoning involves making clinical decisions using various formal and logical techniques. This process is evident primarily in the diagnosis phase. Judgment has come to mean the use of various "intangibles" such as general feelings about the case and past clinical experience, which help the physician to make clinical decisions. These are evident during prognosis and therapy recommendations. Artificial intelligence has attempted to model both of these processes.

There are, however, some aspects of consultations that computers cannot do, such as the physical examination. The physician gains much firsthand information from general appearance, facial expressions, etc., that is inaccessible to the computer. The design of computer consultation systems must, therefore, take this factor into account and offer mechanisms for the representation of these types of information secondhand.

History of Computers in Medicine

The use of computers in medical decision making began in the early 1960s with the implementation of programs that performed well-known types of statistical analyses. These programs focused on the diagnosis aspect of the consultation: They accepted a set of findings and selected one disease from a fixed set, using methods such as pattern recognition through discriminant functions, Bayesian decision theory, and decision tree techniques (Croft, 1972; Nordyke, Kulikowski, & Kulikowski, 1971). Slightly more complex programs performed "sequential diagnosis." Here, when there is not enough information to make a reliable diagnosis, the next patient test (to get more information) is determined by a strategy that selects the "best" test based on three factors: the cost of the test, the danger to the patient, and the amounts of discriminating information needed and made available by the test.

The appeal of using statistical methods is that the resulting decisions are "optimal" according to specified criteria. Unfortunately, these statistical systems proved unsatisfactory. The mathematics that they have been based upon have assumed that the patient has only one disease and that the data are not erroneous. More fundamentally, certain assumptions and simplifications concerning the independence and mutual exclusivity of various disease states that were made in order to make the statistical techniques practical were found to be unjustified. Furthermore, many prior and conditional probabilities required for complete analysis were simply not available.

Since the early 1970s there has been an increasing application of AI techniques to performing medical decision making. Some of the formalisms, techniques, and languages developed in AI were directly applicable to medicine before, but the new understanding of the nature of the task called for new ways of representing knowledge and reasoning. For example, the classical AI problem-solving techniques of state-space search and theorem proving (see Search) were not directly applicable. Consider a simple application of state-space search to the planning of a treatment. If one assumes that the "initial state" is the diseased patient, that the final state is the "healthy patient," and that the "operators" are various drugs, physical therapies, surgical procedures, etc., it would appear that simple search would find a path between the initial and final states. But there are two fundamental problems. First, the initial state, the disease of the patient, is rarely known with certainty. Second, the application of an operator--i.e., a treatment--is not guaranteed to result in an expected state. In order to deal with these problems, methods for representing **inexact knowledge** and for performing **plausible reasoning** have been developed in each of the consultation systems described below.

From the standpoint of AI, medical diagnosis is a **hypothesis formation** (see article C4) problem. The diagnosis task is to use the clinical findings to form a consistent set of disease hypotheses (not to use findings to select one disease from a fixed set of possible diseases). These hypotheses are typically related to one another in various ways. Each existing system exhibits a different approach to this hypothesis formation problem.

The State of the Art

The state of the art in computer-based medical decision making is represented by the programs described in the following articles. These programs are MYCIN (Shortliffe, 1976),

INTERNIST (Pople, 1975), CASNET/GLAUCOMA (Weiss, Kulikowski, & Safir, 1978), PIP (Szolovits & Pauker, 1978), IRIS (Trigoboff & Kulikowski, 1977), and the Digitalis Advisor (Silverman, 1975; Swartout, 1977b). There are now several other programs under development that use the techniques and ideas developed in the above systems. These include PUFF (Feigenbaum, 1977), a pulmonary function program, HODGKINS (Safrans, Desforges, & Tsichlis, 1976), a system for performing diagnostic planning for Hodgkins disease, and HEAD-MED (Heiser, 1977, 1978), a psychopharmacology advisor. During the development of all these programs, certain issues arose concerning the construction of the programs and their acceptance by the medical community. The major issues and the ways in which these were addressed by the individual systems are also described below.

Representation of knowledge. Two distinct types of medical knowledge must be represented: (a) general knowledge of diseases, manifestations, causal mechanisms, etc., and (b) specific knowledge about the patient, the current medical history, the current therapies, etc. The usual representation formalisms of AI--semantic nets (see article Representation.B2), production rules (Representation.B3), frames (Representation.B7), and predicate calculus (Representation.B4)--are not directly applicable because of the inexact nature of medical knowledge. In all of the consultation systems that have been developed, these representations have been augmented, for example, using a numerical way of expressing strength of belief or strength of association. For example, in MYCIN, the medical knowledge is represented as a set of production rules augmented by "certainty factors." These certainty factors express the strength of belief in the conclusion of a rule, given that all of the premises are true. CASNET uses a causal network representation (basically a semantic network with the one relation, CAUSES) where each CAUSES link is qualified by a number that represents the strength of causality. In INTERNIST, a taxonomy of diseases is stored as a huge tree with each node representing a disease. Associated with each disease node is a list of manifestations, with numerical weights reflecting the strength of association between the disease and the manifestation. In PIP, the frame formalism is augmented by numbers that reflect both the strength of belief in a slot filler and the degree to which the frame itself applies to this patient. In IRIS, where the semantic net and production rule formalisms have been combined, a facility for incorporating an arbitrary representation of strength of belief has been included. Finally, a procedural representation is used in the Digitalis Advisors; it contains a mathematical model of the action of digitalis.

Clinical reasoning. Clinical reasoning is based on the ways different pieces of evidence for particular hypotheses are combined. Each system has a different approach to this problem, but most employ the technique of *thresholding*; if the numerical score of a hypothesis exceeds a certain pre-set threshold (defined by the expert physician), then the hypothesis is believed to be true. The clinical reasoning of MYCIN involves determining parameters (e.g., the infections and causative organisms of a patient) using production rules. The premises of a rule are considered true if the combined value of the associated certainty factors exceeds a predefined threshhold. If several rules contribute to a conclusion about a parameter, then their certainty factors are functionally combined to form a composite certainty factor for this conclusion. These confidence-factor combining functions are based on probability theory. In CASNET, a status measure is associated with each state in the causal network. Weights are propagated both in the forward and backward direction depending on disease causality. A state is considered "confirmed" if its status exceeds a specified threshhold. In INTERNIST, disease hypotheses are scored by a procedure that takes account of the strength of association among: (a) the manifestations exhibited by the patient and the disease, (b) the manifestations associated with the disease that are not

present in the patient, (c) and the confirmed diseases causally related to this one. Disease hypotheses are ranked, and the top-ranked diseases are investigated further. When the difference between the scores of the top two disease hypotheses reaches a predefined criterion, the top ranking disease is confirmed. PIP combines two different methods of reasoning: categorical and probabilistic. Categorical decisions are based on logical criteria rather that numerical values. The probabilistic reasoning involves scoring the disease. A frame can be confirmed either on logical or probabilistic criteria. In IRIS, an attempt is made to confirm nodes of a semantic net as being true for the patient. Information is passed between the nodes of the semantic net via sets of production rules associated with the links. These production rules can encode both logical and probabilistic decisions.

Explanation and justification. The explanation and the justification of a system's line of reasoning are important factors for the acceptance of consultation systems by physicians. Explanation involves showing the user the line of reasoning used in making a particular diagnosis; justification is concerned with the medical accuracy and reliability of the knowledge and the reasoning strategies used.

Only two systems currently address the issue of explanation. MYCIN explains a diagnosis by printing out an English version of the chain of rules used. More complex explanation facilities are provided by TEIRESIAS (Davis, 1977), an explanation and knowledge acquisition system developed in the context of MYCIN. The OWL Digitalis Advisor provides English explanations of its reasoning that are generated directly from the OWL code. The detail of the explanation can be controlled by the program. Both INTERNIST and CASNET are able to summarize the consultation by displaying scores of the hypotheses and statuses of states; however, they are unable to explain the methods they used to arrive at these scores.

The issue of justification is a complex one. Both CASNET and MYCIN can cite references to the research literature in support of diagnoses and treatment recommendations. CASNET is able to provide alternative recommendations based on differing expert opinions. At the heart of the justification issue is the accuracy and reliability of the expert's knowledge and whether this knowledge has been accurately captured in the representation formalism. Often medical experts have differing opinions, and it is not clear whether a consensus should be sought or whether the different opinions should all be represented. CASNET and MYCIN have been developed with the collaboration of groups of experts, and the rules typically represent a general consensus of opinion. The other systems were developed with one main expert; so consensus was not an issue.

Validation. Just as the various instruments and drugs used by physicians must be validated, so must consultation programs. So far, CASNET and MYCIN have undergone relatively extensive clinical trials and have been rated as "expert" in their respective domains by human experts. INTERNIST has yet to undergo formal clinical trials, but it is informally rated as an expert in internal medicine. The Digitalis Therapy Advisors have performed well in limited trials.

Acquisition of knowledge. Knowledge acquisition is the transfer of the experts' knowledge and expertise to the program. Currently the only successful way of doing this is through a knowledgeable intermediary, although eventually experts should be able to communicate directly with the consultation program (see Article B on the TEIRESIAS system).

Concluding Remarks

Despite the extensive work that has been done, none of these systems is in routine clinical use. Physicians have not for the most part accepted them. The main reason is that they have yet to satisfy the "indispensability" criterion: They are not indispensable to the practice of medicine and physicians perform adequately without them. The only AI program that is in routine medical use is PUFF, a pulmonary function program, which is used because it saves the physician a lot of time. Constructed using EMYCIN (the MYCIN system with the knowledge of infectious diseases removed), PUFF uses a set of about 55 rules about pulmonary dysfunction. The program suggests treatment recommendations that can be overridden by the physician.

In order for AI programs to make a significant impact on health care, at least in the short term, it appears that PUFF's example should be followed. The ingredients for a successful application in medicine seem to be (a) a careful choice of the medical problem and (b) the cooperation of interested experts. The domain must be narrow and relatively self-contained; the use of the computer should aid, not replace, the physician; and the task should be one that the physician either cannot do or is willing to let a computer do.

To summarize, the main focuses of activity in the area of medical decision making today are: knowledge engineering, the acquisition of knowledge from experts; knowledge representation, for building and maintaining the large medical knowledge bases; strategy design, for reasoning with the medical knowledge; and program designs that feature explanation capabilities, of their reasoning to users.

References

Feigenbaum (1977) gives a short review of this area of research. Most of the work on medical systems is discussed in detail in the AIM Workshop proceedings (AIM, 1975-78). Recent work on some of the important systems is described in a special issue of the Journal of Artificial Intelligence (Sridharan, 1978).

B. TEIRESIAS--Issues in Expert Systems Design

TEIRESIAS is a system for facilitating automatic acquisition and maintenance of the large knowledge bases used by expert systems. Although TEIRESIAS is not itself an application of AI to some domain, it deals with many important issues in expert systems design that are relevant to all of the programs described in this chapter. The system was developed by Randall Davis as part of his doctoral research at the MYCIN project at Stanford, and this article assumes some familiarity with MYCIN's rule-based knowledge representation scheme and its *backward-chaining* control structure (see Article C1). However, the ideas and techniques that TEIRESIAS uses are not necessarily limited to MYCIN's domain of infectious diseases or to the production-rule formalism used by MYCIN.

Knowledge-based Programs

As discussed in the Applications Overview, systems that achieve expert-level performance in problem-solving tasks derive their power from a large store of task-specific knowledge. As a result, the creation and management of large knowledge bases and the development of techniques for the informed use of knowledge are now central problems of AI research. TEIRESIAS was written to explore some of the issues involved in solving these problems.

Most expert programs embody the knowledge of one or more experts in a field, like infectious diseases, and are constructed in consultation with these experts. Typically, the computer scientist *mediates* between the experts and the program he is building to model their expertise. This is a difficult and time-consuming task, because the computer scientist must learn the basics of the field in order to ask good questions about what the program is supposed to do.

TEIRESIAS's goal is to reduce the role of the human intermediary in this task of *knowledge acquisition*, by assisting in the construction and modification of the system's database. The human expert communicates, via TEIRESIAS, with the *performance program* (e.g., MYCIN), so that he can discover, with TEIRESIAS's help, what the performance program is doing and why. TEIRESIAS offers facilities for modifying or adding to the knowledge base to correct errors: Using TEIRESIAS, the human expert can "educate" the program just as he would tutor a human novice who makes mistakes. Ideas about how this "debugging" process is best carried out are at the core of TEIRESIAS's success.

TEIRESIAS also recognizes the inexact, experiential character of the knowledge that is often required for knowledge-based systems and (as examples below will illustrate) offers the expert some assistance in formulating new "chunks of knowledge" of this sort. Another major aim of the system was to provide a mechanism for embodying strategic information. *Meta-rules* (discussed below) are used to direct the use of object-level rules in the knowledge base and to provide a mechanism for encoding problem-solving strategies.

Interactive Transfer of Expertise.

It is an established result that an expert knows more about a field than he is aware, or capable of articulating completely. Thus, asking him a broad question like "Tell me everything

you know about staph-infections" will yield only a fraction of his knowledge. TEIRESIAS's approach is to present the expert with some errors made by an already established, but still incomplete, knowledge-based program and to ask a *focused* question: "What do you know that the program doesn't know, which makes your expert diagnosis different in this case?"

This interaction is called *transfer of expertise*: TEIRESIAS incorporates into the performance program the capabilities of the human expert. TEIRESIAS does not attempt to derive new information on its own but, instead, tries to "listen" as attentively and intelligently as possible, to help the expert augment or modify the knowledge base.

Interactive transfer of expertise between an expert and an expert program begins when the expert identifies an error in the performance of the program and invokes TEIRESIAS to help track down and correct the error. Errors are manifest as program responses that the expert would not have made or as "lines of reasoning" that the expert finds odd, superfluous, or otherwise inappropriate. The first kind of error might be, for example, a wrong conclusion about the identity of a bacteria. On the other hand, the performance program may just ask the expert, during a consultation, a question that, in the expert's opinion, does nothing to resolve the identity of the bacteria. This is an example of the "line of reasoning" type of error.

Both kinds of error are assumed, by TEIRESIAS, to be indicative of a deficit, or "bug," in the performance program's knowledge base. Transfer of expertise begins when TEIRESIAS is called upon to correct the deficit. TEIRESIAS fixes bugs in the knowledge base by:

1. Stopping the performance program when the human expert identifies an error.
2. Working backwards through the steps in the performance program that led to the error, until the bug is found.
3. Helping the expert fix the bug by adding or modifying knowledge.

To identify faulty reasoning steps in the performance program, the expert can use the WHY and HOW commands to ask TEIRESIAS to back up through previous steps, *explaining* why they were taken. The same explanatory abilities can also be used when there is no bug, to help the user follow the system's line of reasoning. Since many large performance programs carry out very complex inferences that are essentially "hidden" from the person using the program, this is a valuable facility.

Meta-level Knowledge

One of the principal problems of AI is the question of appropriate representation and use of knowledge about the world (see Representation). Numerous techniques have been used to represent domain knowledge in various applications programs. A central theme of the research on TEIRESIAS is exploring the use of *meta-knowledge*. Meta-level knowledge is simply the representation in the program of knowledge about the program itself--about how much it knows and how it reasons. This knowledge is represented using the same representation techniques used to represent the domain knowledge, yielding a program containing *object-level* representations describing the external world and *meta-level* representations that describe the internal world of the program, its self-knowledge. For

example, many AI programs use the notion of a *frame* to represent the knowledge used by the system (see Article Representation.B7). One can imagine a meta-level frame that describes the structure of all frames in the system or one that denotes the different classes of frames used in the system. One of TEIRESIAS's representations is very close to this notion, the *schema* described below.

Meta-level knowledge has taken several different forms as its uses have been explored, but it can be summed up as "knowing about what you know." In general, it allows the system both to use its knowledge directly and to examine it, abstract it, and direct its application. The capabilities for explanation, knowledge acquisition, and strategic reasoning in TEIRESIAS inspired the incorporation of explicit meta-level knowledge, and these capabilities are based on the use of that knowledge.

Explanation

There are two important classes of situations where expert systems should be able to explain their behaviour and results. For the user of the system who needs clarification or reassurance about the system's output, the explanation can contribute to the *transparency* and thus the *acceptance* of the system. The second major need for explanation is in the debugging process described above, where a human expert uses the system's explanations of why it has done what it has done, in order to locate some error in the database. The first of these applications of explanation has been explored in the question-answering facility of the MYCIN system; the explanation capability in TEIRESIAS has explored both uses but has concentrated on the latter.

The techniques used in TEIRESIAS for generating explanations are based on two assumptions about the performance program being examined, namely, (a) that a recapitulation of program actions can be an effective explanation, as long as the correct level of detail is chosen, and (2) that there is some shared framework for viewing the program's actions that will make them comprehensible to the user. In the MYCIN-like expert systems that use production-rule knowledge bases, these assumptions are valid, but it is easy to imagine expert systems where one or both are violated. For example, the first assumption simplifies the explanation task considerably, since it means that the solution requires only the ability to record and play back a history of events. This assumption rules out, in particular, any need to simplify those events. However, it is not obvious, for instance, that an appropriate level of detail can always be found. Furthermore, it is not obvious how this approach of recapitulation, which often offers an easily understood explanation in programs that reason symbolically, would be applied to expert systems that perform primarily numeric computations.

A simple recapitulation will be an effective explanation only if the level of descriptive detail is constrained. It must be *detailed* enough that the operations the system cites are comprehensible; the conceptual level must be *high* enough that the operations are meaningful to the observer, so that unnecessary detail is suppressed; and it must be *complete* enough so that the operations cited are sufficient to account for all behavior.

The second assumption concerns the user's comprehension of the expert system's activity, which depends on the fundamental mechanism used by the program and the level at which it is examined. Consider a program that does medical diagnosis using a statistical

approach based on Bayes's Theorem. It is difficult to imagine what explanation of its actions the program could give if it were queried about computed probabilities. No matter what level of detail is chosen, such a program's actions are not (nor were they intended to be) a model of the reasoning process typically employed by physicians. Although they may be an effective way for the computer to solve the diagnosis problems, there is no easy way to interpret these actions in terms that will make them comprehensible to humans unacquainted with the program.

Thus, the lack of mechanisms for simplifying or reinterpreting computation means that TEIRESIAS's approach is basically a first-order solution to the general problem of explanation. But, in the context of a MYCIN-like expert system, for which TEIRESIAS was designed, the simple AND/OR goal tree control structure offers a basis for explanations that typically needs little additional clarification. (The operation of TEIRESIAS's explanation facility is illustrated in the sample protocol at the end of this article.) The invocation of a rule is taken as the fundamental action of the system. This action, within the framework of the goal tree, accounts for enough of the system's operation to make a recapitulation of such actions an acceptable explanation. In terms of the constraints noted earlier, it is sufficiently detailed--the actions performed by a rule in making a conclusion, for instance, correspond closely enough to the normal connotation of that word--that no more detailed explanation is necessary. The explanation is still at a high enough conceptual level that the operations are meaningful and the explanation is complete enough--there are no other mechanisms or sources of information that the observer needs to know in order to understand how the program reached its conclusions.

Knowledge-acquisition: Rule Models and Schemata

When the expert has identified a deficit in the knowledge base of the performance program, TEIRESIAS questions him in order to correct the deficit. This process relies heavily on meta-level knowledge about the performance program, encoded in *rule-models* and *schemata*. In other words, TEIRESIAS knows about what the performance program knows.

The meta-level knowledge about *objects* in the domain includes both structural and organizational information and is specified in *data structure schemata*. Acquisition of knowledge about new objects proceeds as a process of instantiating a schema--creating the required structural components to build the new data structure and then attending to its interrelations with other data structures. By making inquiries in a simple form of English about the values of the schema's components, this knowledge acquisition process is made to appear to the expert as a natural, high-level inquiry about the new concept. The process is, of course, more complex, but the key component is the system's description of its own representation.

TEIRESIAS's *rule models* are empirical generalizations of subsets of rules, indicating commonalities among the rules in that subset. For example, in MYCIN there is a rule model for the subset of rules that conclude affirmatively about *organism category*, indicating that most such rules mention the concepts of *culture site* and *infection type* in their premise. Another rule model notes that those rules that mention *site* and infection *type* in the premise also tend to mention the *portal of entry* of the organism.

This knowledge about the contents of the domain rules is used by TEIRESIAS to build *expectations* about the dialogue. These expectations are used to facilitate the process of

translating the English statements into the performance program's internal representation and to identify information missing from the expert's entry. An example of TEIRESIAS's use of rule models in its knowledge acquisition dialogue is given in the sample protocol below.

Meta-rules and Performance Strategies

In performance programs with sufficiently small knowledge bases (like MYCIN's), exhaustive invocation of the relevant parts of the knowledge base during a consultation is still computationally feasible. In time, however, with the inevitable construction of larger knowledge bases, exhaustive invocation will prove too slow. In anticipation of this eventuality, *meta-rules* are implemented in TEIRESIAS as a means of encoding strategies that can direct the program's actions more selectively than can exhaustive invocation. The following meta-rule is from MYCIN's infectious disease domain:

```
                    METARULE 001

If      1) the infection is a pelvic-abscess, and
        2) there are rules which mention in their
           premise enterobacteriaceae, and
        3) there are rules which mention in their
           premise gram positive rods,

Then    There is suggestive evidence (.4) that the rules
        dealing with enterobacteriaceae should be evoked
        before those dealing with gram positive rods.
```

This rule suggests that since enterobacteriaceae are commonly associated with a pelvic abscess, it is a good idea to try rules about them first, before the less likely rules mentioning gram positive rods. Note that this meta-rule does not refer to specific object-level rules. Instead it specifies certain attributes of the rules it refers to, for example, that they mention in their premise enterobacteriaceae.

An Example: TEIRESIAS in the Context of MYCIN

We will now illustrate TEIRESIAS's operation in affiliation with the MYCIN system (see Article C1), paying particular attention to TEIRESIAS's explanation and knowledge acquisition facilities. MYCIN provides the physician with advice about the diagnosis and drug therapy for bacterial infections. The system asks questions about the patient, the infection, the cultures grown from specimens from the patient, and any organisms (bacterium) growing in the culture. (Typically, of course, the exact identity of the organism is not yet known.)

MYCIN's database is composed of rules that specify a situation (involving information about the patient, culture, and organism) and the conclusions that can be drawn in that situation. For example, to conclude whether a patient suffers from a bacteria in the enterobacteriaceae category, MYCIN invokes Rule 95 of its knowledge base:

```
                    Rule 095

IF   the site of the culture is blood, and
     the gram stain is positive, and
     the portal of entry is gastrointestinal tract, and
     [A-the abdomen is the locus of infection, or
      B-the pelvis is the locus of infection]

THEN there is strongly suggestive evidence that
     enterobacteriaceae is the class of organisms for
     which therapy should cover.
```

MYCIN asks questions of the physician to determine which rules are applicable in the current case--i.e., to determine the "situation." To determine if the organism is of type enterobacteriaceae, MYCIN would inquire whether each clause in Rule 95 were true.

The procedure of working backwards from the conclusion to the situation that would make the conclusion true is called *backward chaining* and is MYCIN's fundamental control structure. Backward chaining can go "several levels deep," meaning that the clauses in a situation that must be true (for the conclusion to be true) might themselves be conclusions of other rules. For example, a condition on the identification of enterobacteriaceae is that the infection be localized in the abdomen or the pelvis; but this condition is itself a conclusion, and it is only true if its conditions (that the culture from these places is recent, and is therapeutically significant) are true. Furthermore, the condition of therapeutic significance has its own conditions, and so on. Here is a protocol showing MYCIN's basic style of interaction, the human expert's responses are prefixed with **:

```
---------Patient 281----------
1) Patient's name:
** Albert Hall
2) Sex:
** Male
3) Age:
** 34
4) Have you been able to obtain positive microbiological
   information about a possible infection of Albert Hall?
** Yes

---------Infection 1-----------
5) What is the infection?
** Primary-Bacteremia
6) Please state when signs and symptoms of infection-1 first appeared:
** 15-Dec-75 12:00
---------Culture 1-------------
7) From what site was the specimen for culture-1 taken
** Blood
8) Please give the date and time when this blood culture was obtained:
** 16-Dec-75 12:00
The first significant organism from this blood culture will be
referred to as:
```

```
---------Organism-1----------
9) Enter the identity of ORGANISM-1
** Unknown
10) Is ORGANISM-1 a rod, or coccus, (etc.):
** Rod
11) The gram-stain of ORGANISM-1:
** Gram negative
12) Did ORGANISM-1 grow in the aerobic bottle?
** Yes
13) Did ORGANISM-1 grow in the anaerobic bottle?
** Yes
                :
                :
                :
My therapy recommendations will be based on the following probable
infection(s) and potential causative organism(s):

INFECTION-1 is PRIMARY-BACTEREMIA
      <Item-1> E.COLI [ORGANISM-1]
      <Item-2> PSEUDOMONAS-AERUGINOSA [ORGANISM-1]
```

FIGURE 1. PART OF A DIALOGUE WITH MYCIN.

In sum, MYCIN constructs a *goal tree* (see Article Search.Problem.Reduction) of questions that must be resolved in the course of the consultation to conclude the identity of a bacteria. When it cannot resolve a question by inference from what it knows already, it asks the respondent to provide an answer. Each node of the goal tree has subnodes that result from the application of a rule:

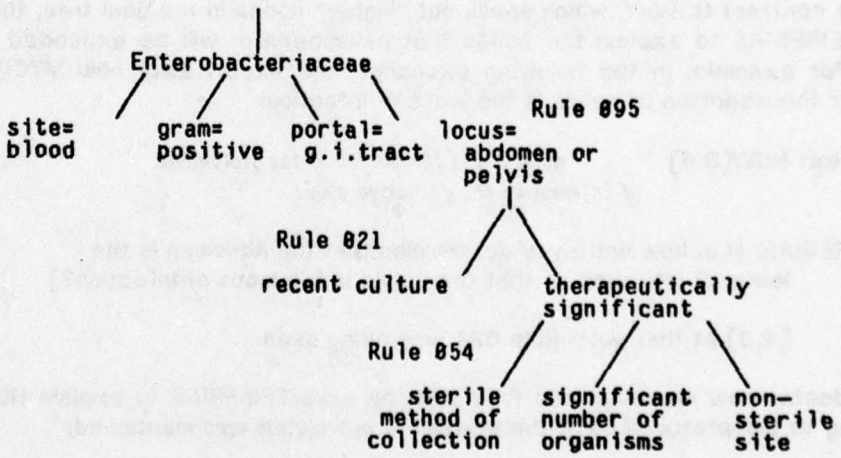


Figure 1. A MYCIN Goal Tree

Explanation: HOW and WHY. Suppose that during a consultation with MYCIN the clinician wants an explanation of the reasoning behind some question that MYCIN has asked him. The explanation given by TEIRESIAS in response to the physician's question, WHY, involves backing up the goal tree to the previous node and presenting the rule used to derive the subnode from its ancestor:

MYCIN: What is the suspected portal of entry of ORGANISM-1 into this sterile site?
Doctor:WHY
TEIRESIAS: [i.e. why is it important to determine the suspected portal of entry of ORGANISM-1 into this sterile site?]

[3.0]...in order to find out about an organism or class of organisms for which therapy should cover.

It has already been established that
[3.1] the site of the culture is blood
[3.2] the stain of the organism is gram positive
Therefore if:
[3.3] the portal of entry of the organism is g.i.
[3.4] [Abdomen is the locus of infection, or Pelvis is the locus of infection]
Then
There is strongly suggestive evidence (.9) that enterobacteriaceae is the class of organisms for which therapy should cover.
[Rule 095]

In contrast to WHY, which spells out "higher" nodes in the goal tree, the HOW command asks TEIRESIAS to explain the nodes that have been or will be expanded lower down the tree. For example, in the following exchange, the expert asks how MYCIN will determine whether the abdomen or pelvis is the locus of infection:

Doctor: HOW[3.4] *note: 3.4 is the prefix of the statement of interest in the exchange above*

TEIRESIAS: [i.e. how were you determining that the Abdomen is the locus of infection, or that the Pelvis is the locus of infection?]

[4.0] At that point Rule 021 was being used.

If the doctor now wants to see Rule 021, he asks TEIRESIAS to explain HOW it was used, referring to the prefix (4.0) of the statement in which it was mentioned:

Doctor: HOW [4.0]

TEIRESIAS: [i.e. how was Rule 021 used?]

[5.0] It has already been established that
[5.1] the culture is recent.

Therefore if:
[5.2] there is therapeutically significant disease associated with the occurrence of this organism
then
it is definite (1.0) that the site of the culture is the locus of infection in the patient.

The explanation facilities of TEIRESIAS involve climbing up and down MYCIN's goal tree explaining WHY questions are asked and HOW they are answered. If MYCIN makes a misdiagnosis, these same facilities are used (either by the expert or TEIRESIAS, at the choice of the expert) to back up through the rules that achieved the diagnosis until the expert has found the faulty one or determined that one is missing. TEIRESIAS is then used to expedite the alteration or construction of rules by the expert.

Knowledge acquisition. TEIRESIAS is able to help an expert formulate rules in English because it has expectations about what rules "look like." The expectations are stored internally in the form of *rule models* that are formed automatically by examining all the rules in the knowledge base. TEIRESIAS knows, for example, that there is a class of rules that have as their conclusions the probable identity of a bacteria and that these rules typically have as conditions (a) the site of the culture, (b) the stain of the culture, and (c) the probable portal of entry of the bacteria. These rule models help TEIRESIAS translate English versions of new rules into a machine representation acceptable to MYCIN.

To illustrate TEIRESIAS's debugging and correction facilities, we will continue with the consultation from Figure 1. Suppose that after MYCIN presents the expert with its diagnosis, TEIRESIAS steps in and checks whether the expert agrees:

MYCIN: My therapy recommendation will be based on the following probable infection(s) and potential causative organism(s):

INFECTION-1 is PRIMARY BACTEREMIA
<Item-1> E.COLI [ORGANISM-1]
<Item-2> PSEUDOMONAS-AERUGINOSA [ORGANISM-1]

Now TEIRESIAS steps in.

Are all these identities correct?

*The expert's response, marked with **, is No.*

** No.
Are there any identities missing?
** Yes.
Which ones?
** KLEBSIELLA-PNEUMONIAE, PROTEUS-NON-MIRABILIS
Are there any identities which shouldn't have appeared?
** yes

Which one(s)?
** PSUEDOMONAS-AERUGINOSA

Do you want to
1] track down the problem yourself, or
2] allow me to guide the debugging, or
3] ignore it for now and fix it another time.
Type 1,2, or 3.

** 2

If the expert had typed 1, he would have used the HOW and WHY commands to debug the problem, as discussed above.

We'll start with PSEUDOMONAS-AERUGINOSA first:

TEIRESIAS starts debugging by trying to find out why an inappropriate diagnosis occurred. Frequently solving a problem will make others "go away" as well.

The following rules were used in deducing that the identity of
ORGANISM-1 is PSEUDOMONAS-AERUGINOSA:

Rule 184

Since
[1.1] the category of ORGANISM-1 is not known
[1.2] the gram stain of ORGANISM-1 is gram negative
[1.3] the morphology of ORGANISM-1 is rod
[1.4] the aerobicity of ORGANISM-1 is facultative

There is weakly suggestive evidence (.3) that the identity of
ORGANISM-1 is PSEUDOMONAS-AERUGINOSA.

Is this rule correct

Note that this rule, which resulted in the wrong conclusion, is not necessarily an incorrect rule. It might be (and in fact is) the case that it was incorrectly invoked because one or more of its conditions are true when they should be false.

** Yes

The expert says the rule is correct.

Should its premise have failed for this case?
** Yes

Which clause of the premise should have failed?
** Clause 1.

The expert believes that MYCIN should have known the category of ORGANISM-1. MYCIN failed to establish the category, causing Rule 184 to be used.

Should it have been possible to conclude a category?
** Yes
Which one?
** Enterobacteriaceae

Our old friend. But recall that MYCIN's rule 095 about enterobacteriaceae requires a positive gram stain, but it can be seen from Fig. 1 that it has a negative gram stain. Thus, Rule 95 was never evoked. In the course of more debugging, it transpires that MYCIN has a couple other rules dealing with this category of bacteria, but they weren't evoked either. The expert decides that another rule is required, and we pick up the dialogue where he is proposing one to TEIRESIAS:

.
.
.

If: 1 ** THE PATIENT'S INFECTION IS PRIMARY-BACTEREMIA
2 ** THE SITE IS ONE OF THE STERILE SITES
Then: ** THERE IS EVIDENCE (.8) THAT THE CATEGORY IS ENTEROBACTERIACEAE

TEIRESIAS parses this rule into its internal representation and then translates it back into English, so the expert can check it. We pick up the dialogue after the expert has agreed on the parse. TEIRESIAS is applying its rule model.

.
.
.

I hate to criticize, Dr. Davis, but did you know that most rules about
what the category of an organism might be, that mention-
the site of a culture
the infection
ALSO mention-
[A]- the portal of entry of the organism
Shall I try to write a clause to account for [A]?
** Yes.

Here we see one of the utilities of the rule model. TEIRESIAS prompts the expert to add a bit of knowledge that is typcially found in rules like these and offers to fix the rule itself. This completion is accomplished by looking at other rules that fit the same rule model, to find the most likely portal of entry clause.

how about -
[A] The portal of entry is gastrointestinal. Ok?
** Yes.

TEIRESIAS now does some finishing up: checking the complete rule with the expert for final approval and asking the expert to write a brief description (for bookkeeping purposes) of why the rule was needed. Finally, it reruns the consultation internally, using the responses from Fig. 1, which it has stored. It turns out that adding the rule above did, in fact, cure the other problems with the first consultation, and this time the diagnosis is satisfactory to the expert.

.
.
.

Summary: TEIRESIAS and Expert Systems

TEIRESIAS aids a human expert in monitoring the performance of a knowledge-based system. When the human expert spots an error in the program's performance, either in the program's conclusions or its "line of reasoning," TEIRESIAS assists in finding the source of the error in the database by *explaining* the program's conclusions--retracing the reasoning steps until the faulty (or missing) rule is identified. At this point, TEIRESIAS assists in *knowledge acquisition*, modifying faulty rules or adding new rules to the database. *Meta-level knowledge* about the kinds of rules and concepts in the database is used to build expectations in TEIRESIAS's *model-based understanding* process. Meta-level knowledge is also used to encode problem-solving *strategies*, in particular, to order the invocation of rules so that those that are most likely to be useful (given the current knowledge of the program) are tried first.

References

The principal reference on TEIRESIAS is the doctoral dissertation by Davis (1976). Uses of meta-knowledge in expert systems are discussed in Davis & Buchanan (1977). Also see Davis (1977) and Davis (1978).

C. Medical Systems

C1. MYCIN

The MYCIN system (Shortliffe, 1976) was designed to provide consultative advice on diagnosis and therapy for infectious diseases. Such advice is often required in hospital care because the attending physician may not be an expert on infectious diseases that show themselves during a period of hospitalization--as, for example, when a cardiology patient develops an infection after heart surgery. Time considerations compound the problem. A specimen (blood, urine, etc.) from a patient can show some early evidence of bacterial growth within 12 hours, but generally it takes 24 to 48 hours (or more) for positive identification. The physician must therefore often decide in the absence of complete information whether or not to start treatment and what drugs to use if treatment is required. Both of these may be difficult questions.

The trace that follows shows the initial and final parts of a sample interaction between a physician and MYCIN (*italicized* comments provide additional commentary but are not part of the actual dialogue). MYCIN, in effect, "interviews" the doctor about his patient, collecting information from which it infers the diagnosis and selects an appropriate therapy.

```
--------PATIENT-1--------
  1) Patient's name: (first-last)
  ** FRED SMITH
```

MYCIN collects some background data about the patient.

```
  2) Sex:
  ** MALE
  3) Age:
  ** 55
  4) Have you been able to obtain positive cultures from a site at which
     Fred Smith has an infection?
  ** YES
     --------INFECTION-1--------
     5) What is the infection?
     ** PRIMARY-BACTEREMIA
     6) Please give the date and approximate time when signs or symptoms of
        the primary-bacteremia (INFECTION-1) first appeared. (mo/da/yr)
     ** MAY 5, 1975
     The most recent positive culture associated with the primary-bacteremia
     (INFECTION-1) will be referred to as:
        --------CULTURE-1--------
        7) From what site was the specimen for CULTURE-1 taken?
        ** BLOOD
        8) Please give the date and time when this blood culture (CULTURE-1)
           was obtained. (mo/da/yr)
        ** MAY 9, 1975
        The first significant organism from this blood culture (CULTURE-1)
        will be referred to as:
           --------ORGANISM-1--------
           9) Enter the identity of ORGANISM-1:
```

** UNKNOWN

Typically, the identity of the organism is not known and hence must be inferred by MYCIN, using various other pieces of laboratory and clinical information. MYCIN proceeds to ask questions that could help to determine the identity of the organism.

10) Is ORGANISM-1 a rod or coccus (etc.):
** ROD
11) The gram stain of ORGANISM-1:
** GRAMNEG
12) Have there been POSITIVE cultures from sites which are NOT associated with infections about which you are seeking advice?
** NO
13) Did ORGANISM-1 grow aerobically?
** YES (.8)
14) Did ORGANISM-1 grow anaerobically?
** YES
15) What is the suspected portal of entry of ORGANISM-1 into this sterile site?
** GI (.6)
...

More questions follow, to gather sufficient information to infer possible identities of the infecting organisms.

My therapy recommendation will be based on the following probable infection(s) and potential causative organism(s):

INFECTION-1 is PRIMARY-BACTEREMIA

<Item 1> The identity of ORGANISM-1 may be PSEUDOMONAS-AERUGINOSA
<Item 2> The identity of ORGANISM-1 may be KLEBSIELLA-PNEUMONIAE
<Item 3> The identity of ORGANISM-1 may be E.COLI
<Item 4> The identity of ORGANISM-1 may be BACTEROIDES-FRAGILIS
<Item 5> The identity of ORGANISM-1 may be ENTEROBACTER
<Item 6> The identity of ORGANISM-1 may be PROTEUS-NON-MIRABILIS

46) On a scale of 0 to 4, where higher numbers indicate increasing severity how would you rate Fred Smith 's degree of sickness?
** 3
47) Does Fred Smith have a clinically significant allergic reaction to any antimicrobial agent?
** NO
48) Fred Smith 's weight in kilograms (or <number> POUNDS):
** 70

Note that 2 drugs cover for 6 potential organisms.

[Rec 1] My preferred therapy recommendation is as follows:
In order to cover for items <1 2 3 5 6>:
Give: GENTAMICIN
Dose: 119 mg (1.7 mg/kg) q8h IV [or IM] for 10 days
Comments: Modify dose in renal failure
In order to cover for item <4>:
Give: CLINDAMYCIN
Dose: 595 mg (8.5 mg/kg) q6h IV [or IM] for 14 days
Comments: If diarrhea or other GI symptoms develop, patient should be evaluated for possible pseudomembranous colitis.

Do you wish to see the next choice therapy?

Alternative choices are available.

** YES
Which, if any, of the above drugs do you wish eliminated from consideration?
** CLINDAMYCIN

[Rec 2] Next best therapy recommendation:
In order to cover for items <2 3 4 5 6>:
Give: CHLORAMPHENICOL
Dose: 1 gm (15 mg/kg) q6h IV for 14 days
Comments: Monitor patient ' s white count
In order to cover for item <1>:
Give: GENTAMICIN
Dose: 119 mg (1.7 mg/kg) q8h IV [or IM] for 10 days
Comments: Modify dose in renal failure

Knowledge Base Organization

The medical knowledge used by MYCIN is encoded as production rules (see article Representation.B3) of the sort shown in Figure 1 below. The rules are stored internally in the INTERLISP form shown, from which the English version is generated. Each rule is a single "chunk" of domain-specific information indicating an action (in this case a conclusion) that is justified if the conditions specified in the premise are fulfilled. Since the rule uses a vocabulary of concepts common to the domain, it forms, by itself, a comprehensible statement of some piece of domain knowledge. As will become clear, this characteristic is useful in many ways.

Each rule is highly stylized--with the if/then format and the specified set of available primitives. While the LISP form of each is executable code (the premise, in fact, is simply EVALuated by LISP to test its truth; and the action, EVALuated to make its conclusions), this tightly structured form makes it possible to examine the rules as well as execute them. For example, the rules can be translated into a readable English format as in Figure 1. This translation capability has been used in MYCIN to explain the program's inferences to the expert. The ability to explain a line of reasoning leading to a conclusion and to justify why the program is asking a particular question in a given case is important. Physicians are more

likely to accept the recommendations of a system that can explain its rationale for making them. This ability is discussed in more detail in Article B on TEIRESIAS.

```
RULE050

If      1) the infection is primary-bacteremia, and
        2) the site of the culture is one of the sterile sites, and
        3) the suspected portal of entry of the organism is the gastro-
           intestinal tract,
then there is suggestive evidence (.7) that the identity of the organism is
     bacteroides.

PREMISE:        ( AND (SAME CNTXT INFECT PRIMARY-BACTEREMIA)
                      (MEMBF CNTXT SITE STERILESITES)
                      (SAME CNTXT PORTAL GI))
ACTION:         (CONCLUDE CNTXT IDENT BACTEROIDES TALLY .7)
```

Figure 1. MYCIN production rule.

The current knowledge base contains 450 such rules that enable MYCIN to diagnose and prescribe therapy for bacteremia (infections of the blood) and meningitis.

Note that the rules are judgmental, that is, they make *inexact inferences* on a confidence scale of -1.0 to 1.0, where -1.0 represents complete confidence that a proposition is false and 1.0 represents complete confidence it is true. In the case of the above rule, the evidence cited in the premise is enough to assert the conclusion shown with a mild degree of confidence: 0.7. This number is called the "certainty factor," or CF, and embodies a model of confirmation described by Shortliffe (1976). MYCIN uses CFs rather than other, more standard statistical measures to decide between alternatives during a consultation session. Standard statistical measures were rejected in favor of CFs because experience with clinicians had shown that they do not use the information comparable to implemented standard statistical methods. However, the concept of CFs did seem to fit the clinicians' reasoning patterns--their judgments of how they weighted factors, strong or weak, in decision making.

The CFs are a measurement of the association between the premise and action clauses of each rule. When a production rule succeeds because its premise clauses are true in the current context, the CFs of the component clauses that indicate how strongly each clause is believed are combined, and the resulting CF is used to modify the CF specified in the action clauses. Thus, if the premise was only weakly believed (low, positive total CF) then any conclusions that the rule might make would be modified (reduced) to reflect this weak belief that the patient was in a particular situation. In questions 13 and 15 in the transcript above, the user shows ack of complete confidence. In addition, since the conclusion of one rule may be the premise of another, reasoning from premises with less-than-complete confidence factors is commonplace.

The premise of each rule is a Boolean combination of one or more *clauses*, each of which is constructed from a *predicate function* with an *associative triple* (attribute, object, value) as its argument. Thus, each premise clause typically has the following four components:

⟨predicate function⟩ ⟨object⟩ ⟨attribute⟩ ⟨value⟩

For example, the second clause in rule050, above, is:

The site of the culture is one of the sterile sites

or, in INTERLISP:

```
(MEMBF        CNTXT      SITE        STERILESITES)
  |             |          |              |
Predicate     Object   Attribute       Value
```

MEMBF is a predicate, and the triple says that the site of the current object (an organism, in this case) is a member of the class of sterile sites. There is a standardized set of some 24 domain-independent predicate functions (e.g., SAME, KNOWN, DEFINITE) and a range of domain-specific attributes (e.g., IDENTITY, SITE), objects (e.g., ORGANISM, CULTURE), and associated values (e.g., E.COLI, BLOOD). These form the "vocabulary" of conceptual primitives available for use when constructing rules.

A rule premise is always a conjunction of clauses, but it may contain arbitrarily complex conjunctions or disjunctions nested within each clause. (Instead of writing rules whose premise would be a disjunction of clauses, a separate rule is written for each clause.) The action part indicates one or more conclusions that can be drawn if the premises are satisfied, making the rules purely inferential.

Medical facts about the patient are represented as 4-tuples made up of an associative triple and its current CF (see Figure 3 below). Positive CFs indicate that the evidence confirms the hypothesis; negative CFs indicate disconfirming evidence.

```
(IDENT ORGANISM-2 KLEBSIELLA .25)
(IDENT ORGANISM-2 E.COLI .73)
(SENSITIVS ORGANISM-1 PENICILLIN -1.0)
(IMMUNOSUPPRESSED PATIENT-1 YES 1.0)
```

Figure 3. MYCIN 4-tuple.

MYCIN's model of inexact reasoning permits the coexistence of several plausible values for a single attribute, if this is suggested by the evidence. For example, after attempting to deduce the identity (IDENT) of an organism, MYCIN may have concluded (correctly) that there is evidence of both E.coli and Klebsiella.

To summarize, there are two major forms of knowledge representation in use in the performance program: (a) the attributes, objects, and values--which form a vocabulary of domain-specific conceptual primitives, and (b) the inference rules expressed in terms of these primitives.

The Inference Engine

In MYCIN, rules are invoked in a simple *backward-chaining* fashion that produces an exhaustive *depth-first* search of an AND/OR goal tree (see article Search.Problem.Reduction). For example, assume that the program is attempting to determine the identity of an infecting

organism. It retrieves all the rules that make a *conclusion* about the topic (i.e., that mention the identity of bacteria in their action) and invokes each one in turn, evaluating each *premise* to see if the conditions specified have been met. For the sample rule above, this process would begin with determining the type of infection. Since the type of the infection is unknown, it is set up as a subgoal and the process recurs.

The search is thus depth-first (because each premise condition is thoroughly explored in turn); the tree that is sprouted is an and/or goal tree (because rules may have OR conditions in their premise); and the search is exhaustive (because the rules are inexact; so that even if one succeeds, it was deemed a wisely conservative strategy to continue to collect all evidence about the subgoal.

The subgoal that is set up is a generalized form of the original goal. Thus, for the first clause in the rule ("the infection is primary-bacteremia"), the subgoal set up is "determine the type of infection." The subgoal is therefore always of the form "determine the value of *attribute*" rather than "determine whether the *attribute* is equal to <value>." By setting up the generalized goal of collecting all evidence about an attribute, the performance program effectively exhausts each subject as it is encountered, and thus tends to group together all questions about a given topic. This feature results in a system that displays a much more focused, methodical approach to the task, which is a distinct advantage where human engineering considerations are important. The cost is the effort of deducing or collecting information that is not strictly necessary. However, since this unnecessary effort occurs rarely--only when the <attribute> can be deduced with certainty to be the <value> named in the original goal--it has not proven to be a problem in practice.

If after trying all relevant rules to resolve a subgoal, the total weight of the evidence about a hypothesis falls between -.2 and .2 (an empirical threshold), the answer is regarded as still unknown. This result would occur if no rules were applicable, if the applicable rules were too weak, if the effects of several rules offset each other, or if there were no rules for this subgoal at all. In any of these cases, when the system is unable to deduce the answer, it asks the user for the value of the subgoal (using a phrase that is stored along with the attribute itself).

This strategy, of always attempting to deduce the value of a subgoal and asking the user only when deduction fails, insures a minimum number of questions. It could also mean, however, that work might be expended searching for a subgoal, arriving perhaps at a less than definite answer when the user might already know the answer with certainty. To prevent this inefficiency, some of the attributes have been labeled "laboratory data," to indicate that they represent information available to the program as results of quantitative tests. In these cases the deduce-then-ask procedure is reversed, and the system will attempt to deduce the answer only if the user cannot supply it. Given the desire to minimize both tree search and the number of questions asked, there is no guaranteed optimal solution to the problem of deciding when to ask for information and when to try to deduce it. But the distinction described has performed quite well and seems to embody an appropriate criterion.

Two other additions to straightforward tree search increase the inference engine's efficiency. First, before the entire list of rules for a subgoal is retrieved, the program attempts to find a sequence of rules that would establish the goal with certainty, based only on what is currently known. Since this is a search for a sequence of rules with CF= 1, the result is termed a *unity path*. Besides efficiency considerations, this process offers the

advantage of allowing the program to make "commonsense" deductions with a minimum of effort (rules with CF = 1 are largely definitional). Because there are few such rules in the system, the search is typically very brief.

Second, the inference engine performs a partial evaluation of rule premises. Since many attributes are found in several rules, the value of one clause (perhaps the last) in a premise may already have been established while the rest are still unknown. If this clause alone would make the premise false, there is clearly no reason to do all the search necessary to establish the others. Each premise is thus "previewed" by evaluating it on the basis of currently available information. The result is a Boolean combination of TRUEs, FALSEs, and UNKNOWNs; and straightforward simplification (e.g., F x U = F) indicates whether the rule is guaranteed to fail.

Therapy Selection

After MYCIN determines the significant infections and the organisms that cause them, it proceeds to recommend an antimicrobial regimen if this is appropriate. The MYCIN therapy selector (Clancey, 1979) uses a description of the patient's infections, causal organisms, a ranking of drugs by sensitivity, and a set of drug *preference categories* (such as "propose 2 drugs: one second choice drug and one third choice drug") to recommend a drug regimen. The algorithm will also modify dosages in the case of renal failure in the patient. The program can provide detailed explanations about how it made a regimen choice and can accept and critique a regimen proposed by the physician.

Acquisition and Use of New Knowledge

The representation of knowledge as production rules and the ability to explain specific rules allow MYCIN to interact with an expert clinician in a manner that permits the system to acquire and use new knowledge. The TEIRESIAS system (see article B; also Davis, 1976) works in conjunction with MYCIN and allows the expert to inspect faulty reasoning chains and then add and modify any rules or clinical parameters required to augment and repair the medical knowledge of MYCIN.

When the expert is dissatisfied with the system's performance on a particular case, MYCIN is able to explain how it made the erroneous conclusions and to guide the expert while he is determining the source of the reasoning "bug." To correct the reasoning, the expert may elect to enter new rules or alter existing ones. The user enters his requests through what is nearly a natural language interface. These requests are parsed and used by the system to create a new internal rule that is then presented to the user for inspection. This interaction helps minimize any misunderstanding between the clinician and MYCIN.

Once this new rule is accepted and understood by the system, the next consultation will make use of it and alter its recommendations accordingly. This ability permits the system to interact directly with the domain experts without intervention of a programmer.

Concluding Remarks

Formal evaluations of the MYCIN system have been done that indicate that MYCIN compares favorably with infectious disease experts in diagnosing and selecting therapy for patients with bacteremia and meningitis. At present, however, the system is not used on the wards, primarily due to its incomplete knowledge of the full spectrum of infectious diseases.

MYCIN is one of the first of a new breed of computer systems: systems that step out of the toy worlds of AI into the real world. These systems must deal with many of the social and psychological problems of *man/machine interactions*. Issues such as modularity and representation of knowledge, reasoning in specific domains, explanation of a system's logic, and the ability to accumulate and use new information must be considered with attention to both programming and interfacing problems. MYCIN has been designed with these issues in mind and has consequently shown promise as a real-world aid to the clinician.

References

See Shortliffe (1976) and Davis (1976).

C2. CASNET

The Causal Associational NETwork (CASNET) program (Weiss, Kulikowski, & Safir, 1977) is a computer system for performing medical diagnosis developed at Rutgers University. The major application of CASNET has been in the domain of glaucoma. The system represents a disease not as a static "state" but as a *dynamic process* that can be modeled as a network of causally linked pathophysiological states. The system diagnoses a patient by determining the pattern of pathophysiological causal pathways present in the patient and identifying this pattern with a *disease category*. Once the disease category is explicitly identified, the most appropriate treatments can be prescribed. The causal model also makes possible a prediction of the likely future course of a disease both if treated and if untreated.

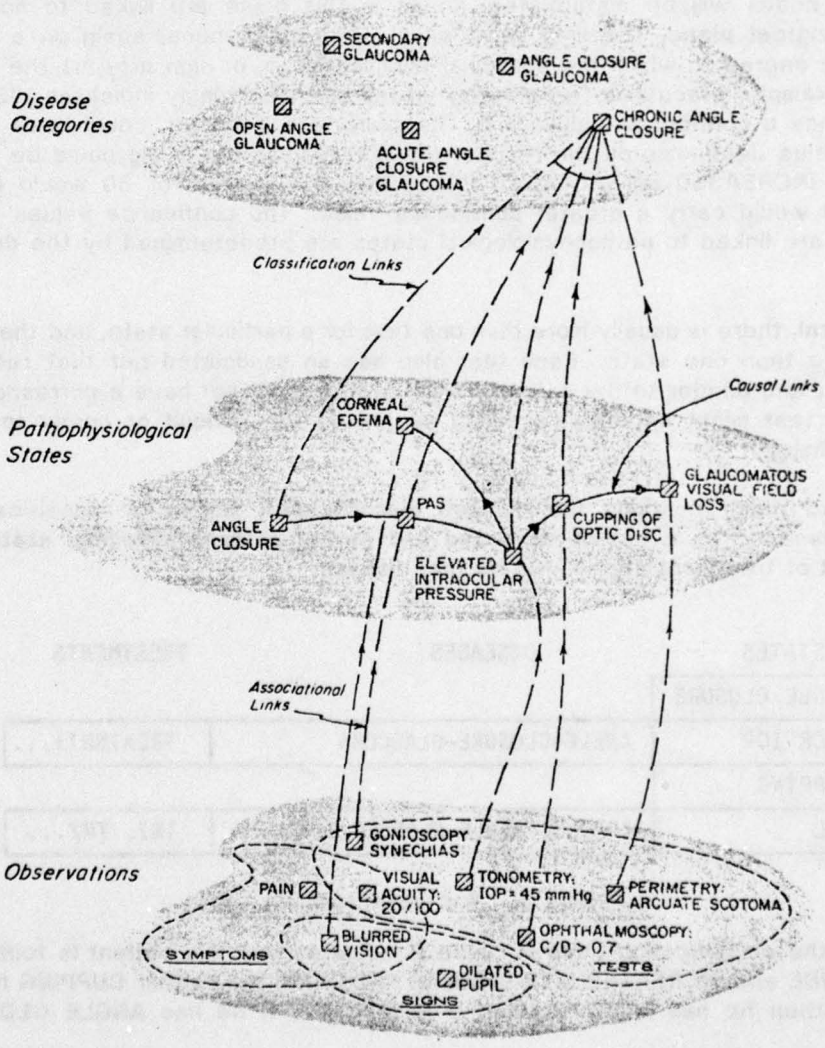


FIG. 1. Three-level description of a disease process.
From Weiss, 1978.

Representation of Medical Knowledge

A CASNET model consists of three "planes of knowledge", parts of which are shown in Figure 1. The plane of pathophysiological states is the heart of the model. The nodes in this plane represent elementary hypotheses about the disease process, and arcs here represent a causal connection between two elementary hypotheses; for example, **INCREASED INTRAOCULAR PRESSURE....CAUSES....CUPPING OF THE OPTIC DISK.** Associated with each link is a forward weight or *confidence factor*, a number on a 1-5 scale, where 5 corresponds to "(almost) always causes" and 1 to "rarely causes." The determination of these weights and their utility in confirming or disconfirming the presence of a pathophysiological state are discussed later in this article.

The **plane of observations** contains nodes representing evidence gathered from the patient. These include signs, symptoms, and laboratory tests. During a consultation, some or all of these nodes will be instantiated. Nodes in this plane are linked to nodes in the **pathophysiological plane.** The links have associated confidences, again on a 1-5 scale, reflecting the degree to which the particular test, symptom, or sign *supports* the associated state. For example, a scotoma (a perimetry measurement) strongly indicates **VISUAL FIELD LOSS**, so it has a confidence value of 5. The same test, however, could have a different confidence value depending on the results; for example, 15 mm of Hg could be considered evidence for **INCREASED INTRAOCULAR PRESSURE**, but a result of 30 would be definite evidence and would carry a greater confidence value. The confidence values with which observations are linked to pathophysiological states are predetermined by the designers of CASNET.

In general, there is usually more than one test for a particular state, and the same test indicates more than one state. Each test also has an associated *cost* that reflects both monetary cost and danger to the patient. Some states might not have a corresponding test since such a test might not exist or might be judged too difficult or costly to use for a particular pathology.

The third plane contains the disease **classification tables.** A classification table defines a "disease" as a set of confirmed and denied pathophysiological states. It also contains a set of treatment statements for that disease.

STATES	DISEASES	TREATMENTS
ANGLE CLOSURE		
INCR IOP	ANGLE-CLOSURE-GLAUCOMA	TREATMNTI...
CUPPING		
VFL	CHRONIC-ANGLE-CLOSURE GLAUCOMA	TR1, TR2...

Figure 2. A classification table.

For example, the classification table in Figure 2 indicates that if a patient is found to have **ANGLE CLOSURE** and **INCREASED INTRAOCULAR PRESSURE** but neither **CUPPING** nor **VISUAL FIELD LOSS**, then he has **ANGLE CLOSURE GLAUCOMA**; if he has **ANGLE CLOSURE** and

INCREASED INTRAOCULAR PRESSURE and CUPPING and VISUAL FIELD LOSS, then he has CHRONIC ANGLE CLOSURE GLAUCOMA. The concept represented in the classification tables is that a disease is dynamic with respect to time and that confirmed states further down a *pathway* represent more advanced stages of the disease. The states in a classification table will generally be on the same pathway. A "starting state" is a state with no causes in the network (also called a basic disease mechanism). Inadequate understanding of disease mechanisms or incomplete models sometimes lead to classification tables containing states from more than one pathway.

Reasoning

Figure 2 illustrates how CASNET defines a disease as a conjunction of causally related pathophysiological states. Diagnosis in CASNET is a matter of finding one or more causal pathways between these states. Reasoning in CASNET is designed to maximize the likelihood of finding these pathways, given a set of signs, symptoms, and test results.

A diagnostic session begins with the program's asking the user (physician) a series of questions about the patient. The physican answers with values for any tests, signs, and symptoms, or he answers UNKNOWN. These values, together with the confidences associated with the tests and the weights associated with the causal arcs, are used to compute a *status*, or confidence factor, for each node in the causal net.

The STATUS of a state is affected both by the results of its associated tests and by the STATUSs of the states around it. For example, if A causes B and B is confirmed by observation, then there is strong evidence for A. A general algorithm is used to propagate these weights on a state, both in the forward direction (i.e., along the direction of the causal link) and in the backward direction. A state is marked *confirmed* if its STATUS is greater than a preset threshold, it is marked *denied* if its STATUS is less than a second threshold, otherwise it is *undetermined*. The program uses a strategy for selecting the next question, based on the cost of the test and on the likelihood that it will lead to the confirmation or denial of a state.

After all available symptoms and findings have been entered and after the STATUS's have been computed, the classification tables are used to determine diagnoses and treatments. The tables are selected to cover all confirmed nodes. The strategy for selecting the tables is to find the starting states for which causal pathways can be generated that reach the largest number of confirmed states without traversing a denied state. This procedure is repeated until all of the confirmed states are covered.

The treatment statements of the selected classification tables are then used to select a therapy for the indicated diseases. Like a state, a treatment has an associated STATUS that is interpreted as its confidence in its success as a treatment. The treatment with the highest STATUS is selected. This assessment is repeated for all selected classification tables. A final algorithm decides whether some treatments are subsumed by others, and then the final treatment recommendations are printed. If desired, a short summary of research justifying the diagnosis and treatment can also be printed. The current glaucoma model contains about 150 states, 350 tests, and 50 classification tables.

Concluding Remarks

CASNET adopts a strictly "bottom-up" approach to the problem of diagnosis, working from the tests, through the causal pathways, to a diagnosis. The separation of medical knowledge (encoded in the causal network) from reasoning strategies (embodied in the program) will make the expansion of the disease model, when new research discoveries are made, a simple matter. The program is continually being tested and updated by a computer-based network of collaborators. The model also provides a convenient way of following the progress of a patient's disease over multiple visits--the causal net can be used to view the disease progression, both forwards and backwards, along the pathways. Although CASNET has been used primarily in the area of glaucoma, the representational scheme and decision-making procedures are applicable to other disease areas that are understood well enough to make the process of disease known. The program's performance has been evaluated by opthalmologists and is considered close to expert level.

References

See Weiss, Kulikowski, & Safir (1978).

C3. INTERNIST

INTERNIST (Pople, 1975; Pople, 1977) is a medical consultation program in the domain of internal medicine developed jointly by H. Pople, a computer scientist, and J. Myers, a specialist in internal medicine, both at the University of Pittsburgh. The program is presented with a list of *manifestations* of disease in a patient (e.g., symptoms, physical signs, laboratory data, and history), and it attempts to form a diagnosis. The diagnosis consists of a list of diseases that would account for the manifestations. Using information presented during the course of the consultation, the program is able to discriminate between competing disease hypotheses. The current version of the program only formulates diagnoses and does not recommend treatments.

One of the major goals of the INTERNIST project has been to model the way clinicians do diagnostic reasoning. The program has been used to explore the way that certain symptoms evoke particular diseases in the mind of the clinician: how hypothesized diseases generate expectations of other symptoms, how a clinician focuses on a particular disease area and temporarily ignores certain other symptoms that he judges irrelevant, and how he decides between competing disease hypotheses.

From the standpoint of computer science, INTERNIST is solving a *theory formation* or *hypothesis formation* problem. Determining a satisfactory diagnosis involves inferring a set of hypotheses to explain the patient data. In INTERNIST, the data are manifestations and the hypotheses are diseases.

Diagnosis in internal medicine is complicated because a patient may suffer from a number of diseases simultaneously. Although some diseases are more likely to be associated than others, the possible combinations are too numerous to encode *a priori*. Pople (1977) suggests that a conservative estimate of this number is 10 to the 40th. Clearly, diagnosis of a set of diseases present in a patient is nontrivial. INTERNIST-I accomplishes this diagnosis by sequentially establishing the diseases that best fit the data. INTERNIST-II is an improvement over its predecessor because it establishes the set of diseases in parallel and therefore avoids some of the annoying artifacts of sequential processing, such as considering a number of incorrect diagnoses before "focusing in" on the correct one.

Overview of INTERNIST-I

For INTERNIST-I a *problem* is defined as a set of mutually exclusive disease hypotheses. If a patient has a number of diseases, INTERNIST-I must solve that number of problems. In brief, INTERNIST-I finds a set of diseases that account for some or all of a set of symptoms, then it picks one disease from the set on the basis of a scoring schema, which is the solution for one of the problems. Then it finds another set of diseases that account for some or all of any remaining symptoms and again picks the most likely of these alternatives. It continues in this manner until all symptoms have been accounted for.

Representation of Medical Knowledge

INTERNIST's knowledge of diseases is organized into a *disease tree*, or taxonomy, using the "form-of" relation (see Fig. 1). For example, Hepatocellular disease is a form of liver disease.

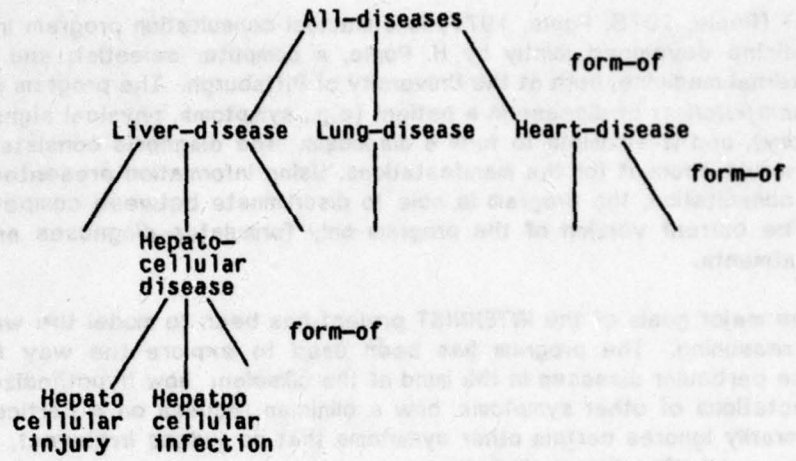


Figure 1. The structure of the disease tree.

The top-level classification in this tree is by organs--heart disease, lung disease, liver disease, etc. A *disease node*'s offspring are refinements of that disease, terminal nodes being individual diseases. A nonterminal node and its subtree are referred to as a *disease area*, while a terminal node is referred to as a *disease entity*. The disease hierarchy is predetermined and fixed in the system.

Diseases and their manifestations are related in two major ways: (a) a manifestation can **evoke** a disease and (b) a disease can **manifest** certain signs and symptoms. These relations can loosely be thought of as probabilities: p(D\M) (the conditional probability of D given M) and p(M\D), respectively. The strength of these relations is given by a number on a 0-5 scale, where 5 means that the manifestation is always associated with the disease and 0 means that no conclusions can be drawn about the disease and the manifestation. Each disease in the tree is associated with its relevant manifestations. Several other kinds of relationships are superimposed on the disease tree to capture causal, temporal, and other association patterns among diseases.

The disease tree and its associated manifestations are constructed and maintained separately from the normal diagnosis program. All known **evokes** and **manifest** relations are entered for the terminal nodes (diseases) of the tree. A list of manifestations is then computed for each nonterminal of the tree by taking the intersection of the manifestation lists of that node's offspring. In this way, the manifestations "percolate" up through the tree to the most general disease with which they are associated and are stored only with this node. This means that manifestations associated with a nonterminal disease node are, by implication, also associated with every node (terminal or non terminal) beneath it in the tree. As well as providing storage economy, this information is used during the consultation for selecting disease areas on which to focus. For example, jaundice (yellowing of the skin) will be associated with some nonterminal disease (e.g., hepatitis) under liver diseases, and its

presence in a patient will cause the consultation program to investigate diseases in that disease area.

Various properties are associated with each manifestation. The most important ones are TYPE and IMPORT. TYPE is a measure of how expensive it is to test for a manifestation, both in terms of financial cost and physical risk to the patient. TYPE is used to order the questions asked by the consultation program: questions about less expensive manifestations are asked first. The IMPORT of a manifestation is a measure of how easily it can be ignored in a diagnosis. The manifestation "Shellfish ingestion" can easily be ignored, but a liver biopsy showing caseating granulomas must be explained.

Reasoning

At the beginning of a consultation, a list of manifestations is entered. As each manifestation is entered, it evokes one or more nodes of the disease tree. A *model* is created for each evoked disease node. The model consists of 4 lists:.

-- Observed manifestations that this disease cannot explain. (This list is called the *shelf*.)

-- Observed manifestations that are consistent with the disease.

-- Manifestations that should be present if this disease is the correct diagnosis but that have not been observed in the patient.

-- Manifestations consistent with this disease but that have not yet been observed in the patient.

After the initial entry of manifestations, the disease tree consists of nodes that have been "lit up" (evoked) and those that have not. A diagnosis corresponds to a set of lit terminal nodes that account for all of the symptoms. In general, at this stage very few of the terminal nodes will be lit up, so the program must ask for further information. To get this further information, the program will focus on a disease area and formulate a problem.

Each disease model is scored, receiving a positive score for each manifestation it explains and a negative score for each manifestation that it cannot explain. Both are weighted by IMPORT. It receives a bonus if it is linked causally to a disease that has already been confirmed. The disease models are partitioned into two sets: (a) the top-ranked model and the diseases that are mutually exclusive to it (alternatives), and (b) the diseases that are complementary to the top-ranked model. For example, if the top-ranked node is hepatocellular injury, then other evoked liver diseases will be alternatives to it, while lung or heart diseases will be complementary.

Having formulated a problem by partitioning the disease models, the system follows one of several strategies, depending on the number of candidate diseases in the problem set. If there are many (more than 4) alternative hypotheses, it attempts to rule out as many as possible. Questions about manifestations that strongly indicate a disease (high (p(M\D)) are selected first. If these manifestations are not present, then this disease can be ruled out. If there are between 2 and 4 possibilities, the program attempts to discriminate between

them. Then questions about manifestations that strongly indicate one disease, D1 (high (p(M\D1))) ,and weakly indicate another disease, D2 (low (p(M\D2))), are selected. These questions are able to discriminate between the two diseases. If there is only one candidate, then questions that have a good chance of confirming this disease are asked. Sometimes, if there is not enough data, it will not be possible to confirm one of the terminal nodes, and a more general diagnosis is given (e.g., "liver disease").

After a disease is confirmed, its manifestations are marked "accounted for"; bonus scores are given to (previously manifested) diseases that are causally linked to this one; and focus shifts to the new top-ranked disease and the formulation of a new problem.

INTERNIST-II

There was a major problem with INTERNIST-I. In complex cases the program had a tendency to begin the analysis by focusing first on totally inappropriate areas. While the final diagnosis was usually correct, the initial meandering was annoying to clinicians. The cause of the problem was traced to the sequential method of problem formulation. The simultaneous formulation of several problems is being investigated in INTERNIST-II.

Representation of Medical Knowledge. INTERNIST-II uses the same database as INTERNIST-I, but it is augmented by a set of *constrictor relations*. These are manifestations that do not evoke a particular disease but, rather, a general area of infirmity. For example, jaundice alerts clinicians to the presence of liver disease. It does not discriminate between liver diseases, but it does delimit this disease area. Formally, a disease area constrained by a constrictor manifestation is a subtree of the disease tree, in this case the subtree of liver diseases.

Reasoning. A problem for INTERNIST-I is to find a set of terminal nodes on the disease tree that accounts for a set of manifestations. It then chooses one node from the set and formulates another problem. INTERNIST-II does not start a diagnosis by formulating a set of terminal nodes, because the number of combinations of terminal disease nodes that may account for a set of manifestations is enormous. Instead, INTERNIST-II partitions the disease tree into disease areas, which collectively account for all the manifestations. Constrictor manifestations are used to make the partitions. If a patient manifests more than one constrictor, then the disease tree will be partitioned into more than one disease area. The conjunction of all the disease areas is called the *root structure* and is formally a set of subtrees of the disease tree. A root structure accounts for all the patient's manifestations. The problem for INTERNIST-II is to decide which terminal nodes (actual diseases) within the root structure best account for the manifestations. This objective is accomplished by partitioning the root structure into smaller subtrees in exactly the same way that the disease tree was partitioned into the root structure, namely, by using manifestations that strongly suggest a disease area, (only this time, the disease area is smaller). The process of partitioning the root structure into smaller areas continues just as long as all the manifestations are accounted for.

This article is a summary account of the operation of INTERNIST-II. In actuality it is more complicated. See Pople (1977) for a complete explication. The main point of INTERNIST-II, however, is that it diagnoses a patient's diseases by dividing the disease tree into smaller and smaller subtrees, until such time as it achieves a set of terminal nodes that accounts for all the manifestations.

Concluding Remarks

INTERNIST I and II have successfully combined a bottom-up and top-down approach to medical diagnosis. The patient data evoke certain disease hypotheses (bottom-up) that are then used to predict (top-down) other manifestations that should be present if the hypothesis is to be confirmed. The system is purely associational. It does not attempt to model any disease processes but considers a disease as a static category and diagnosis as the task of assigning a patient to one or more categories. INTERNIST-I has a large database, currently containing over 500 of the diseases of internal medicine (about 75% complete). It has displayed expert performance in complex cases involving multiple diseases. Pople and Myers expect that the system will be in clinical use in the next few years.

References

See Pople (1975) and Pople (1977).

C4. Present Illness Program

The Present Illness Program (PIP) is being developed at MIT (Pauker et al., 1976; Szolovits & Pauker, 1976; Szolovits & Pauker, 1978). It has been used for taking present illnesses of patients with edemas (accumulation of excess fluids in the body) and patients with renal (kidney) disease. Taking a present illness is different from performing a complete diagnosis. It is the typical consultation, that a patient has with a general practitioner; the patient usually presents a *chief complaint* that becomes the initial focus of the consultation, and only very low-cost sources of information--such as patient history, physical examination, and routine lab tests--are used to make a diagnosis. High-cost or high-risk procedures that may be necessary for a complete diagnosis are not used.

The medical knowledge in PIP is represented as a network of *frames* (see article Representation.B7). The frames are centered around diseases, clinical states, and physiological states (hereafter called the "patient situation") and contain data such as typical findings, relationships to other patient situations, and rules for judging how well a set of *findings* exhibited by a patient "match" the situation described by the frame. *Matching* is the key strategy in the diagnosis. Diagnosis involves matching findings to disease frames and then selecting a set of frames that cover all of the findings. There are, at present, 36 frames for dealing with renal disease.

Currently, the program does not prescribe treatment recommendations. Originally the system was written in CONNIVER (Article AI Languages.D3), but this version was too slow and it has been recoded to run in MACLISP.

Representation of Medical Knowledge

The general medical knowledge in PIP is knowledge about diseases, the patient situation; findings, results of the physical examination and reported symptoms; and the relationships between these entities. This medical knowledge is organized as a *frame system*. Part of a typical frame is shown in Figure 1.

The slots in the frame are grouped into categories as shown. The *typical findings* are those that are expected in a patient having this disorder. However, patients with the disorder need not exhibit all of the typical findings. It is the job of the matching algorithm to compute a "goodness of fit" of findings to a frame. Some of the typical findings have the special status TRIGGER. TRIGGERs are key elements of the clinical decision-making strategy. A TRIGGER is a finding that is sufficiently strongly related to a disorder that presence of the disorder in the patient makes the PIP system attend to the disorder frame as an *active hypothesis*. For example, FACIAL EDEMA is listed above as a TRIGGER for ACUTE GLOMERULONEPHRITIS, meaning that PIP will consider this disease as an active hypothesis if a patient displays facial edema.

```
ACUTE-GLOMERULONEPHRITIS

Typical Findings
   TRIGGERS          (EDEMA with LOCATION=FACIAL.....)
   FINDINGS          (ANOREXIA.....)

Logical Decision Criteria
   IS-SUFFICIENT     (None)
   MUST-HAVE         (None)
   MUST-NOT-HAVE     (None)

Complementary Relations to Other Frames
   CAUSED-BY         (STREPTOCOCCAL-INFECTION, ...)
   CAUSE-OF          (SODIUM-RETENTION, ...)
   COMPLICATED-BY    (ACUTE-RENAL-FAILURE, ...)
   COMPLICATION-OF   (CELLULITIS)

Differential Diagnosis
   CHRONIC-HYPERTENSION implies CHRONIC-GLOMERULONEPHRITIS
   RECURRING-EDEMA implies NEPHROTIC-SYNDROME
   .
   .

Scoring
  (((PATIENT WITH AGE=CHILD) -> 0.8)
   ((PATIENT WITH AGE=MIDDLE-AGED) -> -.6)
    .                              )
  (((EDEMA with SEVERITY = not MASSIVE) -> 0.1)
   ((EDEMA with SEVERITY = MASSIVE) -> -1.0)
    .
    .                              )
```

Figure 1. Part of the frame for acute glomerulonephritis (kidney stones).

The *logical decision criteria* are rules that permit the confirmation or rejection of a hypothesis on the basis of a small number of key findings. Findings strongly correlated with a disease will be listed in the slot IS-SUFFICIENT. If any of these findings are reported, they will be sufficient to confirm the presence of the disease.

The relations between frames reflect the ways in which disorders are related in medicine. Sometimes disease mechanisms are well understood and it is possible to say that one disorder CAUSES another or is a COMPLICATION-OF another. If mechanisms are poorly understood, the disorders may simply be ASSOCIATED. The latter frames are *complementary*, that is, they represent disorders that the patient might have in addition to the initial disorder. In contrast, the *differential diagnosis* slots indicate mutually exclusive disorders--the patient may have one of them and not the disorder represented by the current frame.

The final slot indicates how the findings are scored for the disorder represented by the frame. This score indicates the "goodness of fit" of this disorder to the findings. The statements comprising this slot are sets of clauses that are evaluated in turn. Within a

clause, evaluation terminates when one of the conditions in the clause is true; its score will be used. The local score for a frame is the sum of the values of the clauses, normalized by the maximum total score possible. Thus, 1 denotes complete agreement, while arbitrarily large negative numbers denote complete disagreement.

Reasoning

The clinical strategy used by PIP is based on the manipulations of hypotheses and findings. Knowledge about findings is stored separately from the frame system since a finding can be applicable to many frames. A hypothesis is an instantiation of a disorder frame. There are 3 kinds of hypotheses: (a) confirmed, (b) active, and (c) semi-active. Hypotheses with ratings (as computed by the scoring process) that are higher than a preset threshold are considered *confirmed hypotheses*. *Active hypotheses* are those with at least one confirmed trigger finding; and these contend for the focus of attention. *Semi-active hypotheses* are the immediate neighbors of the active hypotheses in the frame system. They correspond to hypotheses that, although not strong enough to be investigated, are "at the back of the consultant's mind."

The consultation begins with the physician telling the system the main symptoms and signs of a patient. The program then takes the initiative and tries to determine the validity of any active hypotheses by selecting and asking appropriate questions.

The program works through the following cycle:

1. Acquire a new finding. This task is accomplished by asking a sequence of questions that characterizes the finding according to its possible descriptions.

2. Process the finding. All of the frames where this finding is relevant are located.

3. Update the list of active hypotheses. Several kinds of actions can be taken at this point: Remove an active hypothesis if the finding matches a MUST-NOT-HAVE rule; confirm a hypothesis if the premise of an IS-SUFFICIENT rule is now true; activate a hypothesis if the new finding is one of the hypothesis triggers or if the finding allows the premise of a differential diagnosis link to succeed; or revise the score of the hypothesis if the finding matches a scoring rule. If a new hypothesis is activated, then all of its immediate relatives are made into semi-active hypotheses.

4. Select the next finding to query. The highest rated hypothesis becomes the focus of attention, and a question is generated for the next unexplored finding. If there are no hypotheses, a question about a finding for the highest rated causally related frame is asked. Questioning terminates when there are no more active hypotheses or causal relatives with findings to be determined.

If the logical decision criteria are insufficient to confirm or deny a hypothesis, the score of the hypothesis is computed by combining (a) the value of a function that measures the fit of observed findings and typical (expected) findings for the frame (called the *matching score*), and (b) the value of a function that is the ratio of the number of findings

accounted for by the hypothesis to the total number of findings (the *binding score*). The matching score in turn consists of two parts, a local score for the frame (described above) and a score propagated from causally related frames.

Concluding Remarks

Like INTERNIST (see article C3) and unlike MYCIN (article C1), PIP is intended to simulate the clinical reasoning of physicians. The way in which the general medical knowledge has been represented as a system of hypothesized disorder frames and clinical findings reflects this intent, as do the strategies used to select questions for confirming a hypothesis.

The system uses two types of reasoning, *categorical* and *probabilistic*. Decisions about the applicability of a hypothesis are determined using logical decision criteria (the IS-SUFFICIENT, MUST-HAVE, and MUST-NOT-HAVE rules) that a physician uses. When these are insufficient, the probabilistic methods (the computation of matching scores and binding scores) are used. Both kinds of reasoning feature a combination of local and global decision strategies. Local strategies decide how well the findings fit a particular frame, while global strategies determine how well a set of frames fits the findings.

There are a number of difficulties with the program. The questioning can be erratic, since the top-ranked hypotheses tend to alternate rapidly. This oscillation is unlike a physician's line of reasoning, which tends to concentrate on questions that resolve one hypothesis at a time. There is also the problem of when to stop the questioning. The current approach is to stop questioning only when all questions about all possibly relevant hypotheses have been exhausted. This strategy seems too conservative; many irrelevant questions tend to get asked.

References

See Pauker et al. (1976), Szolovits & Pauker (1976), and Szolovits & Pauker (1978).

C5. Digitalis Advisors

There has been considerable work at MIT to develop programs that advise physicians on the administration of the drug digitalis (Silverman, 1975; Swartout, 1977a; and Swartout, 1977b). These programs are not concerned with diagnosing the need for the drug in a patient; rather, they determine an appropriate treatment regimen and its subsequent management for patients known to require digitalis.

Digitalis is administered to patients with erratic heartbeat to stabilize the heart rhythm. The therapeutic effect of digitalis is achieved by maintaining the proper amount of the drug in the bloodstream. The body, however, excretes the drug through the kidneys and liver. Furthermore, overdoses of digitalis are toxic and can cause the very symptoms that the drug is prescribed to cure. A typical digitalis regimen consists of an initial dose that is then modified in response to the effects of the drug on the patient and to the amount of drug being passed by the kidneys.

A mathematical model of the effects of digitalis in the body has existed since 1967. It accounts for the relation betwen the level of body drug stores (as effected by body weight, renal function, etc.) and the incidence of digitalis toxicity. However, application of this model requires that a physician adjust the dosages of digitalis recommended by the model to allow for special sensitivity that a patient might have (or might develop) to the drug. A skilled physician is still required to monitor a patient's progress after the initial dose of digitalis is recommended by the mathematical model.

More recently, Pauker, Szolovits, and their colleagues at MIT have developed a program that makes a model of the effect of digitalis in a *specific* patient and modifies the model in response to feedback about the patient over time (Silverman, 1975). Previously, serum (blood) levels of digitalis had been used to provide feedback, but they proved unsatisfactory alone, and now clinical signs (e.g. nausea, or increased heartbeat irregularity) are used to assess whether the patient is responding well to digitalis, suffering from the toxic effects of the drug, or having no reaction at all.

The combination of a general, mathematical model of the effects of digitalis and a patient-specific model that incorporates clinical data in a continuous feedback cycle has resulted in a Digitalis Advisor system which performs comparably to expert cardiologists in the prescription and maintenance of digitalis therapy.

The Digitalis Advisory system operates as follows:

1. Interact with a physician to build a model of a *specific* patient. This interaction involves asking about the heart condition that digitalis will regulate, about the condition of the kidneys, etc.

2. Prescribe a daily dosage level on the basis of the patient's height and weight and the information gathered in (1).

3. Follow-up sessions assess the patient's response to the dosage recommendations in (2) and modify the dosage if necessary.

In 1977 this program was augmented with the ability to make explanations, but for the sake

of brevity (and because Article B is devoted to this issue) we will not discuss this capability here. See Swartout (1977a) for a complete discussion of the explanation facility.

The Patient-specific Model

What follows is part of an annotated transcript with the Digitalis Advisor, taken from Swartout (1977a).

What is the age of the patient?
Enter numerical value ** 56

What is the weight of the patient?
Units: Kg
Enter numerical value ** 72

What is the sex of the patient?
The possibilities are:
1.Male
2.Female
Enter single value ** MALE

What is the cardiac rhythm?
The possibilities are:
1.Normal sinus rhythm
2.Atrial fibrillation
3.Atrial flutter
4.Pat
5.Pat with block
6.Non-paroxysmal junctional tachycardia
Enter single value ** 2

So far the system is determining what it needs to know to make a patient-specific model. Later, it will ask why the patient is being put on digitalis, whether the patient is currently using the drug, etc.

Is there regularization of ventricular rhythm?
The possibilities are:
1.Present
2.Absent
Enter single value ** WHY?

The user of the system wants an explanation of what motivates a question. The system responds by converting the goals on its goal stack to English, to produce the first two sentences of the explanation. The system then finds and displays a canned explanation which is associated with the concept that is being questioned. The canned explanation usually provides information that could not be deduced from the code of the Advisor:

My top level goal is to begin therapy. I am now trying to check
the cardiac rhythm. If the cardiac rhythm is atrial fibrillation

and there is ventricular-rhythm regularization this suggests that the pacemaking impulses from the ventricles are no longer coming from the atria. If the status of ventricular-rhythm regularization is present it is possible there may be non-paroxysmal junctional tachycardia which is a sign of digitalis toxicity.

The system continues to query the user so that it can build a patient-specific model. Eventually it arrives at the following point:

How long do you wish to take to reach full digitalization?
The possibilities are:
1.Instantaneous rate (1-2 hours)
2.Rapid rate (1 day)
3.Moderate rate (2-3 days)
4.Slow rate (4-7 days)
Enter single value ** 3

Parameters used to make recommendations:
Body store goal: 0.544Mg
Projected amount on board: 0.000Mg
Body stores shift: 0.544Mg
Remaining time to reach goal: 48.0 hours
Half-life: 43.7 hours 1.8 days
Daily loss at equilibrium: 0.175 Mg
Absorbtion factor: 0.75

The system then asks how many times a day the user wishes to administer digitalis and, on the basis of this information, makes dosage recommendations.

The next interaction with the Advisor comes in a follow-up session. The point of this session is for the Advisor to determine whether the patient shows any therapeutic effect of the drug or whether s/he displays any signs of toxicity, and to adjust the dosage accordingly. There are three levels of therapeutic effect: None, Partial, and Complete. Similarly, there are three levels of toxicity: None, Possible, and Definite. There are therefore nine therapeutic/toxic states, and each has an associated set of recommendations associated with it. Digitalis toxicity is identified by five different types of signs and symptoms, including non-cardiac signs (nausea, etc.), and direct cardiac signs of toxicity (e.g., an increase of over 20% in the number of premature ventricular contractions). If any cardiac manifestations are present, the patient is considered definitely toxic; the category "Possibly Toxic" is indicated by various combinations of signs and symptoms from classes other than the cardiac signs.

We will not consider the follow-up session in detail here. See Swartout, 1977a for a complete transcript.

Concluding Remarks

The performance of the Digitalis Advisor reported by (Gorry, Silverman, and Pauker, 1978) suggests that the advisor can perform at least as well as physicians in the

prescription and monitoring of digitalis therapy. In particular, the Advisor was used to make recommendations about therapy for a group of patients who were under the care of house staff in a hospital, and it performed at least as well as the staff, who were themselves under the direction of an attending physician.

References

See Gorry, Silverman, and Pauker (1978) and Silverman (1975). The explanation capability is described by Swartout (1977a).

C6. IRIS

The design goals for IRIS (Trigoboff & Kulikowski, 1977; Trigoboff, 1978) are different from those of the other consultation systems constructed to be expert clinical decision-making systems in a particular medical domain. IRIS was designed to be a tool for building and experimenting with such systems. Developed at Rutgers University and written in INTERLISP, it was designed to permit easy experimentation with alternative representations of general medical knowledge, clinical strategies, and modes of interaction. It was designed to be used by a computer specialist in collaboration with a domain expert. A consultation system for glaucoma has been developed using IRIS.

IRIS uses a combination of two well-established representation formalisms for representing knowledge, *semantic nets* and *production rules* (see articles Representation.B2 and Representation.B3). The semantic net consists of nodes representing patient information and uses a large and extendable set of *link types* for associating this medical knowledge. A set of production rules is associated with each link of the network. The transmission of information between nodes of the semantic network is controlled by the production rules. This process is called *propagation* and is the basis of any clinical strategy implemented in IRIS.

Representation of Medical Knowledge

As with the other medical consultation systems, IRIS makes a (very sharp) distinction between general medical knowledge and any patient-specific knowledge. The general medical knowledge is represented partly as a semantic net and partly as production rules. The nodes of the net represent clinical concepts such as pathophysiological states, diseases, symptoms, findings, treatments, etc. Examples of nodes in the glaucoma application are OPEN ANGLE GLAUCOMA, SCOTOMA, PILOCARPINE THERAPY. The links represent relations between the nodes--e.g., CAUSES, TREATMENT-FOR, SYMPTOM-OF, ASSOCIATED-WITH.

The patient-specific knowledge gathered during a consultation is represented as a set of knowledge structures called "Information SPECifications" (ISPECs). ISPECs are associated with nodes of the semantic net and are created, deleted, and modified during the course of the consultation. An ISPEC is an assertion about the patient and is essentially a frame (see article Representation.B7) with the following slots:

NODE - The name of the associated node in the semantic net. The node represents the concept being asserted about this patient.

SIDE - This slot indicates the half of the body to which this ISPEC refers. Its possible values are LEFT, RIGHT, or NIL. Some nodes in the net will be applicable to a left organ and a right organ (e.g., eye) while others are not (e.g., headache, diabetes). The use of SIDE provides an economical representation, since many nodes might otherwise be duplicated in the net.

MB - This slot is a measure of belief that reflects the degree of system belief in the assertion represented by the ISPEC. Any numerical method of representing degrees of certainty can be implemented here. In the glaucoma application, the confidence factor mechanism of MYCIN (see

article C1) has been implemented. The MB is a pair of numbers: SB (strength of belief) and SD (strength of disbelief). The actual MB is the difference between these two numbers and ranges from total belief to total disbelief.

TIME - The time slot is a list of two dates, the date the ISPEC became true of the patient and the date the ISPEC ceased to be true. The system can also work with a "coarser" view of time: PAST, PAST-OR-PRESENT, and FUTURE. This time representation is part of the mechanism for dealing with multiple visits and for following a patient through a given course of therapy.

MODIFIERS - These are further specifications and qualifications of the basic ISPEC. Examples of modifiers are VALUE, DEGREE, COLOR, and WIDTH. These modifiers do not appear in all ISPECs, but only in those to which they are applicable. These modifiers allow further patient-specific specifications of the concept in the semantic net. For example, "severely increased intraocular pressure" is represented as an ISPEC for INCREASED INTRAOCULAR PRESSURE with modifier, DEGREE: SEVERE.

TYPE - The type slot of an ISPEC determines the way in which it is interpreted. An arbitrary number of types is possible. Currently implemented TYPEs are NIL (the standard and default), FAMILYHISTORY, PATIENTHISTORY, and a number of others that are used by the diagnosis strategy--CHOSEN, COVERED-BY, SUBSUMED-BY, and TREATED-BY.

The statement "The pressure is 10 in the right eye" is equivalent to the ISPEC:

```
NODE = INTRAOCULAR PRESSURE
SIDE = RIGHT
MB   = (1,0)
TIME = PRESENT
MODS = VALUE: 10
TYPE = NIL
```

Reasoning

IRIS makes no committment to any particular strategy of question selection. Currently a "questionnaire" strategy has been implemented. At the beginning of a consultation the program runs through a set of questions and the user answers them.

In the applications of IRIS where consultation and diagnosis are the goal, ISPECs are associated first with the set of symptoms displayed by the patient. In IRIS's knowledge base, symptom nodes are linked to, among other things, disease nodes. Thus, a set of disease nodes can be activated by the symptoms; a disease node is said to explain the symptom nodes that characterize it. Disease nodes are also linked to treatment nodes, and when IRIS has determined which disease(s) holds for a patient, it will activate the appropriate (linked) treatment nodes.

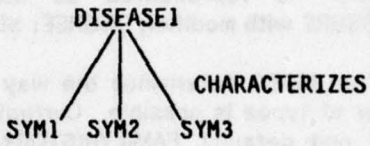
The process of nodes evoking each other in IRIS is called *propagation of ISPECs*, because an ISPEC is associated with a symptom, disease, or treatment node relevant to a

patient. When symptoms evoke a disease or when a disease evokes a treatment, an ISPEC is created. This propagation of information and generation of inferences between any linked nodes in the semantic net is controlled by a set of production rules associated with the link. If the ISPECs associated with the node at the tail of the link satisfy the precondition pattern of a rule, then the actions specified by the rule will be performed at the node at the head of the link. Typical actions include the creation or deletion of ISPECs and the modification of MBs. Thus, IRIS uses a *forward-chaining* reasoning process.

An important propagation pattern is that of the "propagation cone." Consider the rule:

if SYMPTOM1 and SYMPTOM2 and SYMPTOM3 then DISEASE1

In the semantic net, the nodes in this rule would be represented as follows:



Clearly, an ISPEC should only propagate to DISEASE1 if all three symptoms are present. In the case depicted above, propagation should be from the base of the "cone" to the "apex." This propagation pattern is achieved by associating the same decision table with all three CHARACTERIZES links (essentially AND-ing SYM1, SYM2, and SYM3 into one production to insure that ALL symptoms are present before a disease node is evoked). In some cases the direction of propagation will be from apex to base; for example, when propagating "COVERED-BY" ISPECS from a treatment node to each of the diseases it treats.

The production rules are encoded as *decision tables* to make their execution more efficient. Consider the following set of production rules:

R1: if A and B then D
R2: if B and (not C) then (not E)
R3: if A and B and (not C) then F

In evaluating these rules, A and C are evaluated twice and B is evaluated three times. A decision table encoding these three rules is:

	R1	R2	R3
A	+		+
B	+	+	+
C		−	−
	*	*	*
D	+		
E		−	
F			+

A column of the decision table corresponds to a rule. A condition is evaluated only once, and the result is used in each applicable column.

The IRIS claim is that any clinical strategy can be implemented using the available medical primitives. In fact, the propagation of weights in CASNET, therapy selection in MYCIN, and the formation of composite hypotheses in INTERNIST-II were implemented with very little effort (Trigoboff, 1978).

Clinical strategy of IRIS for glaucoma diagnosis

The clinical strategy for the glaucoma application is implemented via a set of 6 special nodes in the semantic net: CHOSEN-DIAGNOSIS, CHOSEN-TREATMENT, POSSIBLE-DIAGNOSIS, POSSIBLE-TREATMENT, UNEXPLAINED-SYMPTOM, and UNTREATED-PATHOLOGY. The goal of the consultation is (a) to have one or more ISPECs associated with the nodes CHOSEN-DIAGNOSIS and CHOSEN-TREATMENT, and (b) to have all ISPECs associated with UNEXPLAINED-SYMPTOMS and UNTREATED-PATHOLOGY be TYPE=COVERED-BY. As findings are entered, they propagate ISPECs to the node UNEXPLAINED-SYMPTOMS. Propagation across SYMPTOM-OF links will result in ISPECs with varying CFs (confidence factors), associated with a number of disease nodes. Any disease with a high enough CF will propagate an ISPEC to the node POSSIBLE-DIAGNOSIS. After all data has been entered, the diseases associated with POSSIBLE-DIAGNOSIS are then investigated in turn. Each diagnosis temporarily receives TYPE=CHOSEN, and TYPE=COVERED-BY propagates to each symptom explained by this disease. The number of explained symptoms is used as a measure of the explanatory power of a disease. This process, of temporary assignment, is repeated for each possible diagnosis; and the disease that explains the most symptoms is given a permanent TYPE=CHOSEN. If there are any unexplained symptoms, the process is repeated.

A similar strategy using the nodes POSSIBLE-TREATMENT, CHOSEN-TREATMENT, and UNTREATED-PATHOLOGY is used to select treatments.

Concluding Remarks

IRIS has been explained in the context of its glaucoma application, but it was designed to represent medical knowledge from ANY domain and to implement a variety of clinical strategies. (Recall that aspects of CASNET, MYCIN, and INTERNIST-II have all been implemented in IRIS.)

This generality is feasible because the representation of knowledge is itself very general (augmented semantic nets). In principle, knowledge from any (medical or nonmedical) domain can be represented. A second characteristic of IRIS that makes it very general is the separation of clinical strategy, both conceptually and operationally, from medical knowledge. Note that to implement the "consultation" strategy, IRIS needed to "know about" only six nodes in the knowledge base: chosen diagnosis, chosen treatment, possible diagnosis, possible treatment, unexplained symptom, and untreated pathology. These six concepts are inherent to the clinical strategy of consultation; every other node in the knowledge base is conceptually and operationally independent of the implementation of the clinical strategy.

References

See Trigoboff & Kulikowski (1977) and Trigoboff (1978).

References

AIM Workshop Proceedings. Dept. of Computer Science, Rutgers University. Held annually, 1975-78.

Clancey, W. Tutoring rules for guiding a case method dialogue. **International Journal of Man-Machine Studies**, 1979, 11, 25-49.

Croft, J. Is Computerized Diagnosis Possible? **Computers and Biomedical Research**, 1972, 5(4), 351-367.

Davis, R. **Applications of Meta-Level Knowledge to the Construction, Maintenance and Use of Large Knowledge Bases**, Stanford AI Lab Memo AIM-283, AI Lab, Stanford University, 1976.

Davis, R. Interactive transfer of expertise: Acquisition of new inference rules. IJCAI 5, 1977, 321-328.

Davis, R. Knowledge acquisition in rule-based systems: Knowledge about representations as a basis for system construction and maintenance. In D. Waterman & F. Hayes-Roth (Eds.), **Pattern-directed Inference Systems**. New York: Academic Press, 1978. Pp. 99-134.

Davis, R., & Buchanan, B. Meta-level knowledge: Overview and Applications. IJCAI 5, 1977, 920-928.

Davis, R., Buchanan, B., & Shortliffe, E. H. Production Rules as a Representation for a Knowledge-base Consultation Program. **Journal of Artificial Intelligence**, 1977, 8(1), 15-45.

Feigenbaum, E. A. The art of artificial intelligence: Themes and case studies in knowledge engineering. IJCAI 5, 1977, 1014-1029.

Feinstein, A. **Clinical Judgment**. Baltimore: William & Wilkins, 1967.

Gorry, A., & Barnett, O. Sequential diagnosis by computer. **Journal of the American Medical Association**, 1968, 205, 849-854.

Gorry, G. A., Silverman, H., and Pauker, S. G. Capturing clinical expertise: A computer program that consdiers clinical response to digitalis. *American J. of Medicine*, 1978, 64, 452-460.

Heiser, J. A computerized Psychopharmacology Advisor. **HEAD-MED Report in the SUMEX Annual Report**. Computer Science Dept., Stanford University, 1977-1978.

Jacquez, J. A. **The Diagnostic Process**. Ann Arbor, Mich.: Mallory Lithography, 1964.

Ledley, R., & Lusted, L. Reasoning foundations of medical diagnosis. **Science**, 1959, 130(3366), 9-21.

Nordyke, R., Kulikowski, C. A., & Kulikowski, C. W. A Comparison of Methods for the Automated Diagnosis of Thyroid Dysfunction. Computers and Biomedical Research, 1971, 4(4), 374-389.

Pauker, S., Gorry, A., Kassirer, J., & Schwartz, W. Towards the Simulation of Clinical Cognition--Taking a Present Illness by Computer. American Journal of Medicine, June 1976, 60, 981-996.

Pople, H. E. The DIALOG Model of Diagnostic Logic and Its Use in Internal Medicine. IJCAI 4, Tbilisi, USSR, 1975.

Pople, H. E. The formation of composite hypotheses in diagnostic problem solving--an exercise in synthetic reasoning. IJCAI 5, 1977, 1030-1037.

Safrans, C., Desforges, J., & Tsichlis, P. Diagnostic Planning and Cancer Management, MIT/LCS/TR-169, MIT, 1976.

Shortliffe, E. H. Computer-Based Medical Consultations: MYCIN. New York: Elsevier, 1976.

Silverman, H. A Digitalis Therapy Advisor, MAC TR-143, Computer Science Dept., MIT, 1975.

Sridharan, N. S. Journal of AI: Special issue on applications in the sciences and medicine, 1978, 11(1,2), 1-195.

Swartout, W. A Digitalis Therapy Advisor with Explanations, MAC TR-176, Computer Science Dept., MIT, 1977. (a)

Swartout, W. A Digitalis Therapy Advisor with Explanations. IJCAI 5, 1977, 819-825. (b)

Szolovits, P., & Pauker, S. Research on a Medical Consultation Program for Taking the Present Illness. Proc. 3rd Illinois Conf. on Medical Information Systems, November 1976.

Szolovits, P., & Pauker, S. Categorical and Probabilistic Reasoning in Medical Diagnosis. Journal of Artificial Intelligence, 1978, 10. In press.

Trigoboff, M. IRIS: A Framework for the construction of Clinical Consultation Systems. Doctoral dissertation, Dept. of Computer Science, Rutgers University, 1978.

Trigoboff, M., & Kulikowski, C. IRIS: A System for the Propagation of Inferences in a Semantic Net. IJCAI 5, 1977, 274-280.

Weiss, S., Kulikowski, C., & Safir, A. A Model-Based Consultation System for the Long-Term Management of Glaucoma. IJCAI 5, 1977, 826-832.

Weiss, S., Kulikowski, C., & Safir, A. A Model-Based Method for Computer-Aided Medical Decision-Making. Journal of AI, 1978, 11(1,2), 145-172.

Index