Stanford Heuristic Programming Project
Memo HPP-77-1

Computer Science Department
Report No. STAN-CS-77-593

February 1977

EXPLANATION CAPABILITIES OF PRODUCTION-BASED CONSULTATION SYSTEMS

by

A. Carlisle Scott, William J. Clancey, Randall Davis & Edward H. Shortliffe
Knowledge-Based Consultation Systems Group

COMPUTER SCIENCE DEPARTMENT
School of Humanities and Sciences
STANFORD UNIVERSITY

DAHC15-73-C-0435

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: STAN-CS-77-593, HPP-77-1
2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle): Explanation Capabilities of Production-Based Consultation Systems.

5. TYPE OF REPORT & PERIOD COVERED: Technical Report

6. PERFORMING ORG. REPORT NUMBER: 77-1

7. AUTHOR(s): A. Carlisle Scott, Randall Davis, William Clancey, Edward Shortliffe

8. CONTRACT OR GRANT NUMBER(s): BHSRE Grant HS-01544; ARPA Order-2494; DAHC15-73-C-0435; ARPA Order-2494

9. PERFORMING ORGANIZATION NAME AND ADDRESS: Departments of Computer Science and Clinical Pharmacology, Stanford University, Stanford, Ca 94305

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS: Bureau of Health Sciences Research and Evaluation/ NIH, Bethesda, MD; and ARPA IPT Office, 1400 Wilson Ave, Arlington, VA 22209

12. REPORT DATE: February 1977

13. NUMBER OF PAGES: 30 (35 p.)

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Philip Surra, Resident Representative, Office of Naval Research, Dyrand 165, Stanford University

15. SECURITY CLASS. (of this report): unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Reproduction in whole or in part is permitted for any purpose of the U.S. Government.

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

18. SUPPLEMENTARY NOTES

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

production rules
computer-based consultants
knowledge-based systems
explanation
question-answering
judgmental knowledge
natural language understanding

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

A computer program that models an expert in a given domain is more likely to be accepted by experts in that domain, and by non-experts seeking its advice, if the system can explain its actions. An explanation capability not only adds to the system's credibility, but also enables the non-expert user to learn from it. Furthermore, clear explanations allow an expert to check the system's "reasoning", possibly discovering the need for refinements and additions to the system's knowledge base. In a developing system, an

(cont on p 1473 B)

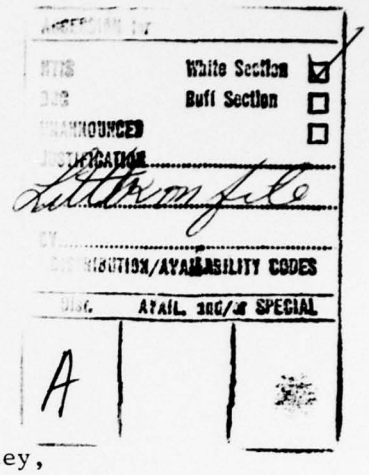
DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

(cont fr p 1473A)

explanation capability can be used as a debugging aid to verify that additions to the system are working as they should.

This paper discusses the general characteristics of explanation systems: what types of explanations they should be able to give, what types of knowledge will be needed in order to give these explanations, and how this knowledge might be organized. The explanation facility in MYCIN is discussed as an illustration of how the various problems might be approached.



EXPLANATION CAPABILITIES OF PRODUCTION-BASED CONSULTATION SYSTEMS

STAN-77-593
Heuristic Programming Project Memo 77-1

A. Carlisle Scott, Randall Davis, William Clancey,
and Edward H. Shortliffe
Knowledge-Based Consultation Systems Group

ABSTRACT

A computer program that models an expert in a given domain is more likely to be accepted by experts in that domain, and by non-experts seeking its advice, if the system can explain its actions. An explanation capability not only adds to the system's credibility, but also enables the non-expert user to learn from it. Furthermore, clear explanations allow an expert to check the system's "reasoning", possibly discovering the need for refinements and additions to the system's knowledge base. In a developing system, an explanation capability can be used as a debugging aid to verify that additions to the system are working as they should.

This paper discusses the general characteristics of explanation systems: what types of explanations they should be able to give, what types of knowledge will be needed in order to give these explanations, and how this knowledge might be organized. The explanation facility in MYCIN is discussed as an illustration of how the various problems might be approached.

KEY WORDS

PRODUCTION RULES, COMPUTER-BASED CONSULTANTS, KNOWLEDGE-BASED SYSTEMS, EXPLANATION, QUESTION-ANSWERING, JUDGMENTAL KNOWLEDGE, NATURAL LANGUAGE UNDERSTANDING

This research was support in part by the Bureau of Health Sciences Research and Evaluation under Grant HS-01544, and by the Defense Advanced Research Projects Agency under ARPA Order No.2494, Contract No. DAHC 15-73-C-0435.

Table of Contents

1 General Discussion

1.1 Consultative Production Systems

A consultation program plays the role of an expert consultant in some domain, giving advice or answers to non-experts with problems in the domain. Users will often want to know how the system arrived at its results during a particular consultation. This paper explains how the implementation of such a program as a production system can facilitate program-generated explanations.

A production system [2] consists of three basic components: a set of production rules, a data base which is both used and updated by these rules, and a rule interpreter. A production rule often is in the form of a situation-action rule: it describes a situation and a set of actions to be taken if this situation is found to exist. The rule interpreter determines the order in which rules will be tried, checks to see if the situations exist, and undertakes the required actions. It also determines how many of the potentially useful rules will be used: only the first (where ordering may be predetermined or computed dynamically), all possible rules, or enough rules to satisfy some criterion that the interpreter uses.

In some production systems, rules are always tried in a predetermined order. In others, the order in which rules are tried varies with different consultations, since a rule will be tried as soon as the rule interpreter determines that it may be useful. In such systems, the common alternatives are data-directed rule invocation, in which a rule is considered "useful" if its situation part matches the data base, and goal-directed rule invocation, in which a rule is "useful" if its action part will help the system reach its current goal. Many systems use a combination of goal- and data-directed rule invocation.

A consultative production system need not be a psychological model, imitating a human's reasoning process. The important point is that the system and a human expert use the same (or similar) knowledge about the domain to arrive at the same answer to a given problem. The system's rules and data base can be viewed as a *knowledge base* containing the domain-specific knowledge of an expert as well as facts about a particular problem. When a rule is used, its actions make changes to the data base which are the system's *decisions* or *deductions*. Thus, a rule can be thought of as a piece of *judgmental knowledge*, using the judgment and knowledge of an expert to make deductions.

The process of trying rules and taking actions can be thought of as "reasoning", and explanations consist of showing how rules used information provided by the user to make various intermediate deductions and finally to arrive at the answer. If the information contained in these rules is sufficient to show why an action was taken (without getting into programming details), an explanation can consist of printing each rule that was used (or an English equivalent of what the rule means.)

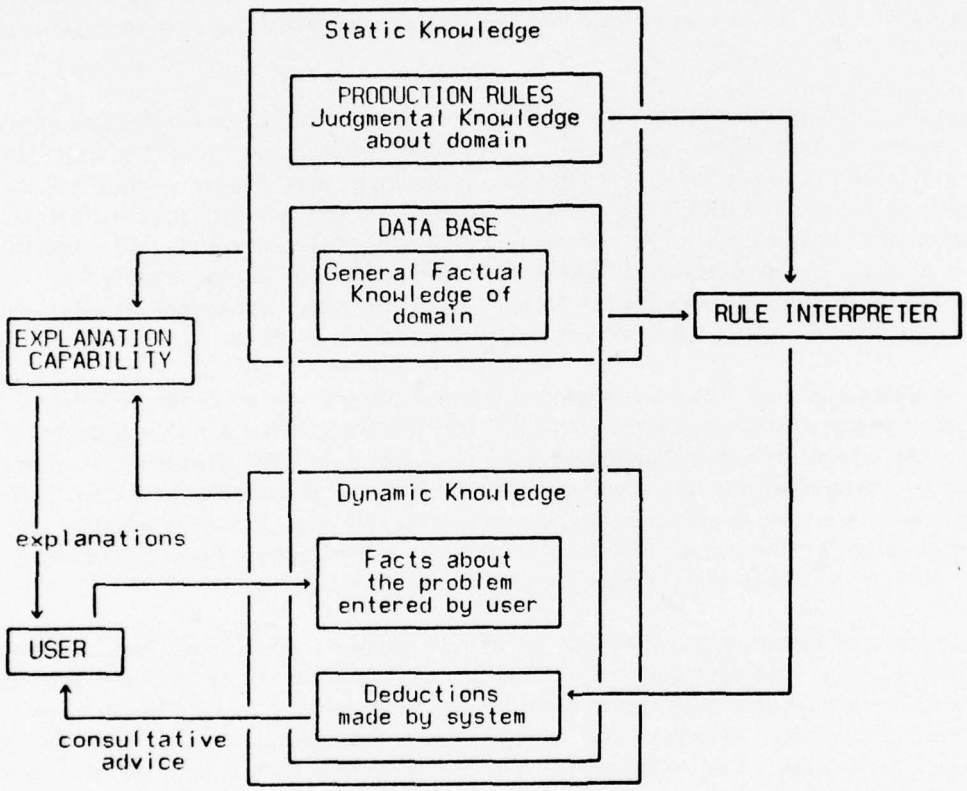


Figure 1. A Production-Based Consultation System with Explanation Capability

The three components of a production system (a RULE INTERPRETER, a set of PRODUCTION RULES, and a DATA BASE) are augmented by an EXPLANATION CAPABILITY. The data base is made up of general facts about the system's domain of expertise, facts that the user enters about a specific problem, and deductions made about the problem by the system's rules. These deductions form the basis of the system's consultative advice.

The explanation capability makes use of the system's knowledge base to give the user explanations. This knowledge base is made up of static domain-specific knowledge (both factual and judgmental) and dynamic knowledge specific to a particular problem.

1.2 Performance Characteristics of an Explanation Capability

The purpose of an explanation capability (EC) is to give the user access to as much of the system's knowledge as possible. Ideally, it should be easy for a user to get a complete, understandable answer to any sort of question about the system's knowledge and operation -- both in general, and with reference to a particular consultation. This implies three major goals in the development of an explanation capability:

1) To ensure that the EC can handle questions about all relevant aspects of the system's knowledge and actions. It should be capable of giving a few basic types of explanations, for example:
   - How it made a certain decision
   - How it used a piece of information
   - What decision it made about some subproblem
   - Why it didn't use a certain piece of information
   - Why it failed to make a certain decision
   - Why it required a certain piece of information
   - Why it didn't require a certain piece of information
   - How it will find out a certain piece of information [while the consultation is in progress]
   - What the system is currently doing? [while the consultation is in progress]

   The specific set of explanation types which are chosen as basics, however, will depend on the particular system.

2) To enable the user to get an explanation which answers the question completely and comprehensively.

3) To make the EC easy to use. A novice should be able to use the EC without first spending a large amount of time learning how to request explanations.

We will distinguish two slightly different functions for an EC, and divide it into two components: the reasoning-status checker (RSC) to be used during the consultation, and the general question answerer (GQA) to be used during the consultation or after the system has printed its results.

A reasoning-status checker will answer questions asked during a consultation about the status of the system's reasoning process. A few simple commands are often sufficient to handle the questions that the RSC is expected to answer.

A general question-answer will answer questions about the current state of the system's knowledge base, including both static domain knowledge, and facts accumulated during the consultation. A GQA will often need the ability to recognize a wide range of question types about many aspects of the system's knowledge. For this reason, it might be difficult to define a few simple commands which would be easy to learn and still cover all the possible questions that might be asked. Consequently, natural-language processing in this component may be important to an explanation system's acceptability.

In an interactive consultation, the system periodically requests information about the problem. This offers the user an opportunity to request explanations while the consultation is in progress. In non- interactive consultations, the user has no opportunity to interact with the system until after it has printed its conclusions. Unless there is some mechanism allowing a user to interrupt the reasoning process and ask questions, the explanation capability for

such a system will be limited to questions about the system's final knowledge state. It will have no reasoning-status checker, and its general question-answerer will only be accessible at the termination of the consultation.

1.3 Knowledge Requirements of an Explanation Capability

An EC must know what is in the system's knowledge base, and how it is organized. In order to give explanations of the system's current (or previous) actions, an EC also needs to understand how the system's rule interpreter works: when rules will be tried, how they can fail, what causes the interpreter to try one rule but not another, etc. This general "schema" for how or why certain rules are used, together with a comprehensive record of the specific actions taken during a particular consultation, can be used as a basis for explaining the results of that consultation.

A reasoning-status checker will need a record of what the system has done so far in order to explain how it arrived at the current step. General knowledge of how the rule interpreter works is necessary in order to explain where the current step will lead. The ability to understand individual rules also may be necessary to the extent that the content of a rule may explain why it was necessary to use this rule, or may affect which future rules will be tried.

A general question-answerer will need more information about the system since the scope of its explanations is much broader: its task is to answer general questions about the system's knowledge base. To do this, it must know how the system stores knowledge about its area of expertise (the static knowledge with which it starts each consultation) and how it stores facts gathered during a particular consultation (its dynamic knowledge). These two types of information will allow a GQA to answer questions about the substance and extent of the production system's current knowledge.

If an explanation capability also is to provide information about how the system arrived at the facts that are currently in its dynamic knowledge base, the GQA will need all the information that a reasoning-status checker uses: a detailed record of the consultation, an understanding of the rule interpreter, and the ability to understand rules.

These three types of knowledge could be supplemented with a limited amount of general information about such things as elementary logic, set theory, and arithmetic comparisons. This would allow the GQA to answer more complicated questions about why the system's knowledge base is in its current state, and to answer questions involving relationships between different facts in the knowledge base.

The nature of the consultation domain, as well as what primary purpose the explanation capability is to serve, will influence the range of questions that an EC should handle. In some systems, a simple retrieval of facts may suffice, while others may need to give detailed description of the production system's "decision" process and to make a number of deductions from facts that it has.

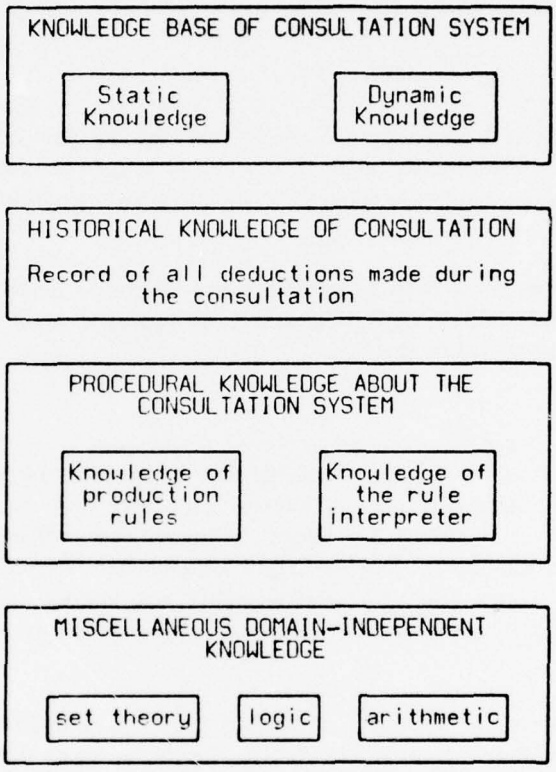


Figure 2. Knowledge Requirements of an Explanation Capability

Access to the consultation system's knowledge base is a prerequisite for performance of the explanation capability. Other types of knowledge may be added to the system to enable the EC to answer a wider range of questions.

1.4 Program Design Considerations

The last two sections described what an explanation capability is, outlining what tasks it should perform, and what it requires in order to perform these tasks. In this section, we discuss design considerations for the parent production system that will enable its EC to meet the requirements that were outlined in the previous section. This discussion is not meant to define the "correct" way of representing or organizing knowledge, but rather to mention certain factors which should be taken into account when deciding what representation or organization will be best for a given production system.

1.4.1 Question Types

The first step is to decide what basic types of questions the system should be able to answer. This will have a direct influence on how the EC is implemented. It is important, however, to make the initial design flexible enough to accomodate possible future additions to the set of basics.

If the basic forms are diverse enough, some level of natural-language understanding may be necessary. The degree of sophistication of the natural-language processor will depend upon what kind of performance is expected of the EC.

1.4.2 Organization of Knowledge

The format and organization of various components of the production system's knowledge base will affect the design of an EC. Individual pieces of static and dynamic knowledge presumably will be organized in some fashion which makes them accessible during the consultation. A GQA facility could make use of such organization to help in finding the information needed to answer a question. The less organized the knowledge base, the more difficult will be the task of the EC, as more complicated routines must be used in order to find the desired information.

During the course of the consultation, the system should keep a record of its actions for use by both components of the explanation capability. Where the ordering of events is important (e.g. when the action of one rule establishes the situation necessary for a subsequent rule to succeed), the record should be structured in a manner which reflects the ordering of events as well as the reasons why each event occurred.

1.4.3 Knowledge of What Rules Mean

The explanation capability will need to understand some of the semantics of individual production rules. This requirement could be met by having the system's knowledge base include a description of what each rule means, encoded in some form which would be of use to the EC. If the format of the system's rules is highly stylized and well-defined, however, it might be possible instead to implement a mechanism for "reading" the rules: the language in which the rules themselves are written could be defined. A high-level description of the individual components of this language, telling what each component means, could be used to enable the EC to read and understand rules. If the rule set consists of a large number of rules, and these rules are composed entirely of a relatively small number of primitive elements, this second approach has the advantage that less information needs to be stored -- a description of each of the primitive components, as opposed to a description of each rule. When new rules are added to the system, the first approach requires that descriptions of these rules must be added. With the second approach, provided that the new rules are

made up of the standard rule components, no additional descriptive information would be needed by the explanation capability.

1.4.4 Knowledge of the Rule Interpreter

Enabling an EC to understand how the rule interpreter works is analogous to enabling it to understand rules. It must be able to "read" the interpreter or else it must have access to some stored description of how the interpreter works. There is a third approach for understanding the rule interpreter, one which would not be feasible for understanding a large number of rules. Knowledge of how the interpreter works could be built into the EC -- the information would not be stated explicitly, but would be used implicitly by the programmer in writing the actual code for the explanation capability. The EC can be thought of as a number of "specialists", each capable of giving a single type of explanation. There could be one specialist for each of the basic question types that the system can answer. Each of the specialists needs only a small amount of information about the rule interpreter which could be built into its "explaining" program.

1.4.5 Other Domain-Independent Knowledge

The final type of knowledge that some general question-answering facilities will need is information allowing deductions to be made from facts in the knowledge base. The representation and extent of this knowledge will depend upon the types of questions that the system is to answer. If logic is needed only to determine the answers to questions of a certain type, for example, the necessary deductions could be built into the specialist for answering that type of question. On the other hand, in some explanation capibilities,the GQA will be expanded to do more than simply give explanations of the system's actions or to query its data base -- it will be expected to answer a wide range of questions involving various kinds of inferences about the knowledge base. Such a GQA will need to check for equality or set membership, make arithmetic comparisons, or make logical deductions. In general, most information of this type can be embodied in a new kind of specialist which is an expert at some sort of logical deduction or comparison. Representation of this sort of general knowledge will become important as the GQA becomes not simply an explanation tool, but also a deductive one.

2 An Example -- MYCIN

2.1 Overview

MYCIN [5,6,7] is an example of a production-based consultation system with a well-developed explanation capability. A production run is an infectious disease therapy consultation in which MYCIN is the infectious disease expert, and the user is a doctor who wants advice about the treatment of a patient.

Knowledge that is gathered during the consultation is organized into attribute-object-value triples. In response to questions during the consultation, the user enters information about the existence of several objects, called *contexts*: the patient, infections that the patient has, organisms which may be causing these infections, cultures that were taken, and drugs that were given. The task of the consultation system is to determine the values of various attributes (called *clinical parameters*) of these contexts. For example, AGE is a clinical parameter of the patient; IDENTITY is a clinical parameter of an organism, with STREPTOCOCCUS as a possible value; SITE is a parameter of a culture, with BLOOD as a possible value.

A clinical parameter's value may be determined by asking the user, or by using decision rules. The parameter is said to be *traced* when the system has done all it can to find out the parameter's value. Tracing a parameter involves asking the user for a value (where applicable) and trying rules for determining the value of that parameter. Rules are tried until the value is known with certainty or there are no rules left to use.

Each decision rule has a situation part called its PREMISE. This consists of predicates, conditions that are tested to determine whether the indicated situation exists. If the conditions in a rule's PREMISE are true, its ACTION will be evaluated, giving new (or updated) values to some parameter(s). Before a condition in a rule's PREMISE can be tested, the parameters that it mentions must be traced. For example, before rule 209 (below) can succeed, the system must know the site of the culture, the portal of entry of the organism, and whether the patient is a compromised host. If any of the clauses in the PREMISE is false, or if the system is unable to find out the value of one of these parameters, the rule will fail.

```
        RULE209
        -------

(PREMISE) If:  1)  The site of the culture is blood, and
               2)  The portal of entry of the organism is GI, and
               3)  The patient is a compromised host
(ACTION)  Then:  It is definite (1.0) that bacteroides is an organism
                 for which therapy should cover
```

Associated with each attribute-object-value triple is a *certainty factor* -- a number between -1 and 1 inclusive which indicates how strongly the system believes that the attribute of the object has the indicated value. The user may modify the answer to any question with a certainty factor, and all rules make conclusions which specify a degree of certainty as well as attribute, object, and value.

Each context is named uniquely, allowing the system to refer to CULTURE-2, meaning the second culture, or ORGANISM-3, meaning the third organism. Moreover, the contexts are organized into a tree known as the *context tree*, which defines relationships among them. For

example, an organism is the direct descendent of the culture from which it was isolated. In the portion of a tree shown in Figure 3, ORGANISM-3 hangs under CULTURE-2 indicating that STREPTOCOCCUS was isolated from the BLOOD culture.

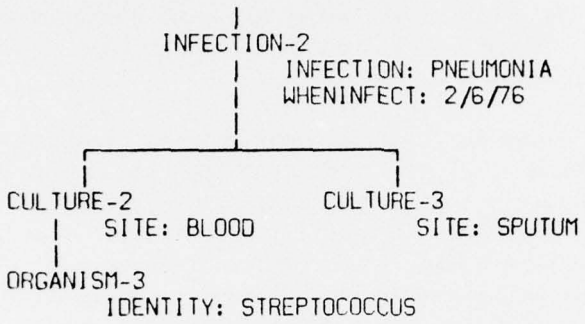


Figure 3. Portion of a Context Tree Showing Some Contexts, Clinical Parameters, and Values

The rule interpreter (MYCIN's control structure, described in detail in [7]) chooses the rules which should be used in the particular consultation, interprets these rules, and creates a record of its actions for use by the explanation system. Rules are invoked to find out values of parameters in a given context. A rule is *applied* to the lowest context in the context tree whose parameters are mentioned by the rule. The rule can use (or conclude about) parameters of this context, or of any context which is its ancestor in the tree. For example, if RULE209 were applied to ORGANISM-3 (see Figure 3) it would need the SITE of the culture from which the STREPTOCOCCUS was isolated. The tree indicates that this is CULTURE-2.

Rather than being a sequential cycle through the rule set, where each rule is tried in some predetermined order, the flow of control is goal-directed. This means that only rules which conclude about the current goal (to find out the value of a given parameter) are examined. The PREMISE of one of these rules may need to use some parameter whose value is unknown. This sets up a subgoal, namely to determine the value of this parameter so that the rule can be used. MYCIN's goal-directed approach means that the system (and not the user) takes the initiative during a consultation. The user will be asked about only those parameters which may be relevant to the particular patient's case.

2.2 *Organization of Knowledge in* MYCIN

In order to give explanations of a consultation system's decisions, an explanation capability must have access to the system's knowledge base. More informative explanations can be given if the EC also has knowledge of how the system works, a record of the consultation, and possibly some domain-independent knowledge. This section discusses how MYCIN meets these requirements.

The system's knowledge base consists of static medical knowledge plus dynamic knowledge about a specific consultation. Static knowledge is further classified as *factual* and *judgmental*. Factual knowledge consists of facts which are medically valid independent of

the particular case. Judgemental knowledge consists of production rules representing deductions which might be made, conditional on what is already known about the case. The format of production rules and of dynamic knowledge has already been described.

2.2.1 Organization of Factual Knowledge

As discussed in Section 2.1, all knowledge which is gathered during the consultation is organized into attribute-object-value triples. For consistency, many facts in the static knowledge base also have this format. This includes objects such as bacteria and antibiotics, and attributes such as the staining characteristics of a bacterium or the recommended dosage of an antibiotic:

```
ATTRIBUTE       OBJECT          VALUE
---------       ------          -----
GRAM            E.COLI          GRAMNEG
DOSE            GENTAMICIN      1.7 mg/kg q8h IV (or IM)
```

The remainder of the factual knowledge consists of lists and tables: pieces of medical knowledge, organized in such a way that they can be used to augment the production rules. For example, one such piece of knowledge is the list of the possible culture sites which are normally nonsterile.

```
NONSTERILESITES:  (CERVIX CUTANEOUS-ULCER LOCHIA NOSE SKIN
                   STOOL THROAT URETHRA VAGINA)
```

The likely pathogens associated with the different culture sites are organized in a table, with different entries for the different sites.

```
PATH-FLORA
----------
THROAT:     (STREPTOCOCCUS-PNEUMONIAE STREPTOCOCCUS-GROUP-A
             NEISSERIA-MENINGITIDIS)
URINE:      (E.COLI PSEUDOMONAS ENTEROCOCCUS PROTEUS KLEBSIELLA
             ENTEROBACTER)
SKIN:       (STAPHYLOCOCCUS-COAG-POS STREPTOCOCCUS-GROUP-A
             STAPHYLOCOCCUS-COAG-NEG)
CERVIX:     (STREPTOCOCCUS CLOSTRIDIUM-GANGRENE NEISSERIA-GONORRHEA
             STREPTOCOCCUS-GROUP-A)
```

Production rules can make use of this tabularized information:

```
RULE058
-------
  If:  1)  The site of the culture is one of: those sites that are
           normally nonsterile, and
       2)  This organism and at least one of the likely pathogens
           associated with the site of the culture agree with respect
           to the following properties: gram morph air
  Then:  There is strongly suggestive evidence (.9) that each of these
         pathogens is the identity of the organism
```

Note that the information in the table could have been organized as attribute-object-value triples (where the object would be a culture site). If this had been done, however, the above

rule could not have been written. To accomplish the same purpose (without a change in the control structure), the system would have needed several rules -- a separate one for each entry in the table. Structuring certain facts into lists and tables enables individual production rules to express general theories which allow a number of specific deductions to be made.

2.2.2 Procedural Knowledge

Each of MYCIN's approximately 400 rules is composed of a small number of conceptual primitives. A total of 60 such primitives make up the language in which rules are written. This design facilitated the implementation of a mechanism for translating rules into English (described in detail in [7]). Each primitive functions has a translation template with blanks to be filled in with translations of the function's arguments. A large part of MYCIN's explanation capability depends on this ability to translate rules into a form that the user can understand.

Having a small number of rule components also facilitates the examination of rules to see which might be applicable to the explanation at hand. MYCIN's knowledge of production rules, therefore, takes the form of a general mechanism for "reading" rules. On the other hand, no attempt has been made to read the code of the rule interpreter. Procedural knowledge about the interpreter is embodied in "specialists", each capable of answering a single type of question. Each specialist knows how the relevant part of the control structure works and what pieces of knowledge it uses.

In order to understand rules, the system's various specialists use a small amount of knowledge about rules in general, together with descriptions or templates of each of the rule components. As an example, the following rule is composed of the units $AND, SAME, and CONCLUDE.

```
RULE009
-------
PREMISE:   ($AND (SAME CNTXT GRAM GRAMNEG)
                 (SAME CNTXT MORPH COCCUS))
ACTION:    (CONCLUDE CNTXT IDENTITY NEISSERIA TALLY 800)

[Translation:
  If:  1)  The gram stain of the organism is gramneg, and
       2)  The morphology of the organism is coccus
  Then:  There is strongly suggestive (.8) that the identity
           of the organism is Neisseria]
```

[When the rule is used, the LISP atom CNTXT is bound to some object, the context to which the rule is applied (see Section 2.1)]

The template for CONCLUDE is shown below. This describes each of the arguments to the function: first, an object (context); second, an attribute (clinical parameter); third, a value for this parameter; fourth, the tally or degree of certainty of the PREMISE; and last, the certainty factor -- a measure of how strong our belief in this conclusion would be, assuming that the PREMISE of the rule is definitely true.

```
CONCLUDE
--------
   TEMPLATE:   (CNTXT PARM VALU TALLY CF)
```

To illustrate how this is used, consider an explanation that involves finding all rules which could conclude that the identity of an organism is Neisseria. The appropriate specialist would start with those rules which the system uses to conclude values for the parameter IDENTITY. Using templates of the various ACTION functions which appear in each of these rules, the specialist picks out only those (like RULE009) which have NEISSERIA in their VALU slot.

This also illustrates the sort of knowledge that can be built into a specialist. The specialist knew that the control structure uses stored lists telling which rules can be used to determine the value of each parameter. Furthermore, it knew that it was necessary to look only at the rules' ACTIONs because it is the ACTION that concludes facts, while the PREMISE uses facts.

2.2.3 The History Tree

Many of the explanation capability's specialists need a record of the consultation. This record is built during the consultation, and is organized into a tree structure called the *history tree* which reflects MYCIN's goal-directed approach. Each node in the tree represents a goal and contains information about how the system tried to accomplish this goal: by asking the user or by trying rules. Associated with each rule is a record of whether the rule succeeded, and if not, why it failed. If trying some rule causes the system to trace a new parameter, thereby setting up a subgoal, the node for this subgoal is the offspring of the node containing the rule which caused the tracing. Figure 4 illustrates how part of a history tree might look. In this example, RULE003 caused tracing of the parameter CATEGORY which is used in the PREMISE of this rule.

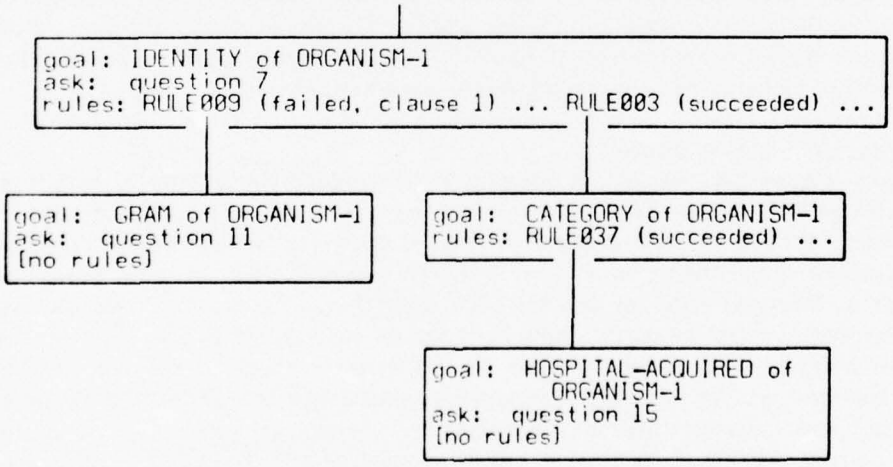


Figure 4. Portion of a History Tree

[RULE009 is shown above, see Figure 5 for RULE003 and RULE037]

2.2.4 Other Domain-Independent Knowledge

MYCIN's question-answering ability is limited to describing the system's actions, and explaining what facts the system knows. Some of the specialists for answering questions about the consultation make use of logic in arriving at their answers. In particular, to explain why a decision wasn't made, the appropriate specialist uses the logical conclusion that the answer consists of explaining what *prevented* the system from using each of the rules that *would have made* that decision.

If deductions or comparisons are needed to answer questions of a specific type, then the necessary logic is built into the appropriate specialist. There is no general representation of knowledge about logic, arithmetic, or set theory that the explanation capability can use to make inferences from different facts in its knowledge base. To find out whether ORGANISM-1 and ORGANISM-2 have the same identity, for example, it is necessary for the user to ask separately for the identity of each organism, then to compare the answers to these questions.

2.3 Scope of MYCIN's Explanation Capability

The purpose of the explanation system is to enable a user to see how MYCIN makes decisions, both in general and with reference to a particular consultation. To make this facility as useful as possible, we have tried to anticipate all types of questions which a user might ask, and to make every part of the system's knowledge base and reasoning process accessible through clear explanations.

The entire explanation facility consists of a number of components or "specialists" each capable of giving a single type of explanation. These components are grouped into three sets: one for explaining what the system is doing at a given time, one for answering questions about the system's static knowledge base, and one for answering questions about the dynamic knowledge base. The first set forms MYCIN's reasoning-status checker; the second and third together make up the system's general question-answer.

2.3.1 MYCIN's Reasoning-Status Checker

Whenever MYCIN asks a question, the user is allowed to interrogate the status of MYCIN's reasoning chain by asking WHY this piece of information is important. As explained in sections 2.1 and 2.2, the system asks a question in order to find out about its current goal. Consider the portion of a history tree shown in Figure 4. HOSPITAL-ACQUIRED is one subgoal, CATEGORY is another at the next level up, and RULE037 links them. The "reason" for asking whether the infection was hospital-acquired, then, is based on an attempt to use this rule to determine the higher subgoal. The answer to WHY at this point is simply a display of the goals and the rules linking them. We have given considerable attention to making this as lucid a display as possible. First, since "why" has several other interpretations, and no more extensive natural language processing is used in this component of MYCIN's EC, the question is expanded to indicate the sense in which it will be answered. The higher goal is then presented, followed by the rule. The clauses in the PREMISE of the rule are divided into those already established and those yet to be determined. Finally, since rules may have multiple conclusions about different clinical parameters, the relevant conclusion is presented first and all others follow.

As Figure 5 illustrates, further sections of the reasoning chain can be examined by repeating the WHY command. For any of the subgoals mentioned in answer to a WHY, the user may ask

HOW this goal was (or will be) achieved. MYCIN's reasoning-status checker is described in more detail in [3,4,6].

```
        .
        .   [preceded by the first 14 questions in the consultation]
        .
15) Is the patient's illness with ORGANISM-1 a hospital-acquired infection?
** WHY
[i.e. WHY is it important to determine whether or not the infection with
     ORGANISM-1 was acquired while the patient was hospitalized?]
[1.0] This will aid in determining the category of ORGANISM-1.
     It has already been established that
        [1.1] the gram stain of ORGANISM-1 is gramneg, and
        [1.2] the morphology of ORGANISM-1 is rod, and
        [1.3] the aerobicity of ORGANISM-1 is facultative
     Therefore, if
        [1.4] the infection with ORGANISM-1 was not acquired while the
              patient was hospitalized
     then
        there is strongly suggestive evidence (.8) that the category of
              ORGANISM-1 is enterobacteriaceae
     ALSO: there is weakly suggestive evidence (.1) that the identity of
              ORGANISM-1 is pseudomonas-aeruginosa]
     [RULE037]
** WHY
[i.e. WHY is it important to determine the category of ORGANISM-1?]
[2.0]... in order to determine the identity of ORGANISM-1.
     It has already been established that
        [2.1] this blood culture was taken from a sterile source
     Therefore, if
        [2.2] this current organism and at least one of the list of members
              associated with the category of the organism agree with
              respect to the following properties: air conformation
      then
        There is strongly suggestive evidence (.9) that each of them is the
              identity of ORGANISM-1
     [RULE003]
** HOW 1.3
[i.e. HOW was it established that the aerobicity of ORGANISM-1 is
     facultative?]
The following were used:
[3.1] RULE027 ndicated there is strongly suggestive evidence (.8) that the
     aerobicity of ORGANISM-1 is facultative
[3.2] RULE027 indicated there is weakly suggestive evidence (.2) that the
     aerobicity of ORGANISM-1 is anaerobic
Since this gave a cumulative CF of (.8) for facultative, and (.2) for
     anaerobic, it has been established that the aerobicity of ORGANISM-1
     is facultative
** HOW 1.1
[i.e. HOW was it established that the gram stain of ORGANISM-1 is gramneg?]
You said so [question 11].
```

Figure 5. MYCIN's Reasoning-Status Checker
[user entries follow the double asterisks]

2.3.2 MYCIN's General Question Answerer

The question-answering part of the system has natural-language routines for analyzing the user's input. The system recognizes questions phrased in a number of ways, thereby making the question-answering facility easier to use. Questions about the static knowledge base may deal with judgmental knowledge (e.g., which rules use or conclude a certain piece of information) or they may ask about factual knowledge -- entries in tables and lists. Some questions about static knowledge are shown in Figure 6.

```
IS BLOOD A STERILE SITE?
WHAT ARE THE NONSTERILE SITES?
WHAT ORGANISMS ARE LIKELY TO BE FOUND IN THE THROAT?
IS BACTEROIDES AEROBIC?
WHAT METHODS OF COLLECTING SPUTUM CULTURES DO YOU CONSIDER?
WHAT DOSAGE OF STREPTOMYCIN DO YOU GENERALLY RECOMMEND?
HOW DO YOU DECIDE THAT AN ORGANISM MIGHT BE STREPTOCOCCUS?
WHY DO YOU ASK WHETHER THE PATIENT HAS A FEVER OF UNKNOWN ORIGIN?
WHAT DRUGS WOULD YOU CONSIDER TO TREAT E.COLI?
HOW DO YOU USE THE SITE OF THE CULTURE TO DECIDE AN ORGANISM'S IDENTITY?
```

Figure 6. Sample Questions about MYCIN's Static Knowledge

Perhaps the more important part of the question-answering system is its ability to answer questions about a particular consultation. While some users may be interested in checking the extent of MYCIN's static knowledge, most questions will ask for a justification of, or for the rationale behind, particular decisions which were made during the consultation. Outlined in Figure 7 are the types of questions about dynamic knowledge which can be handled at present. A few examples of each type are given. <Cntxt> indicates some context which was discussed in the consultation; <parm> is some clinical parameter of this context; <rule> is one of the system's decision rules.

```
1)  what is <parm> of <cntxt>
      TO WHAT CLASS DOES ORGANISM-1 BELONG?
      IS ORGANISM-1 CORYNEBACTERIUM-NON-DIPHTHERIAE?

2)  how do you know the value of <parm> of <cntxt>
      HOW DO YOU KNOW THAT CULTURE-1 WAS FROM A STERILE SOURCE?
      DID YOU CONSIDER THAT ORGANISM-1 MIGHT BE A BACTEROIDES?
      WHY DON'T YOU THINK THAT THE SITE OF CULTURE-1 IS URINE?
      WHY DID YOU RULE OUT STREPTOCOCCUS AS A POSSIBILITY FOR ORGANISM-1?

3)  how did you use <parm> of <cntxt>
      DID YOU CONSIDER THE FACT THAT PATIENT-1 IS A COMPROMISED HOST?
      HOW DID YOU USE THE AEROBICITY OF ORGANISM-1?

4)  why didn't you find out about <parm> of <cntxt>
      DID YOU FIND OUT ABOUT THE CBC ASSOCIATED WITH CULTURE-1?
      WHY DIDN'T YOU NEED TO KNOW WHETHER ORGANISM-1 IS A CONTAMINANT?

5)  what did <rule> tell you about <cntxt>
      HOW WAS RULE 178 HELPFUL WHEN YOU WERE CONSIDERING ORGANISM-1?
      DID RULE 116 TELL YOU ANYTHING ABOUT INFECTION-1?
      WHY DIDN'T YOU USE RULE 189 FOR ORGANISM-2?
```

Figure 7. Sample Questions about a Consultation

Before a question can be answered, it must be classified as belonging to one of these groups. As Figure 7 illustrates, each question type includes a variety of ways in which the question can be worded, some specifying the parameter's value, some phrased in the negative, and so forth. MYCIN's natural-language processor must classify the questions, then determine what clinical parameters, etc. the question references.

2.4 Understanding The Question

The main emphasis in the development of the MYCIN system has been the creation of a production system which can provide sound diagnostic and therapeutic advice in the field of infectious disease. The explanation system was included in the system's original design in order to make the consultation program's decisions acceptable, justifiable, and instructive. Since the question-answering facility was not the primary focus of the research, it is not designed to be a sophisticated natural-language understander. Rather, it uses crude techniques, relying strongly on the very specific vocabulary of the domain, to "understand" what information is being requested.

The analysis of a question is broken into three phases: the first creates a list of *terminal* or root words; the second determines what type of question is being asked (see the classification of questions in Section 2.3); and the last determines what particular parameters, lists, etc. are relevant to the question.

In the first and last steps, the system dictionary is important. The dictionary contains approximately 1400 words that are commonly used in the domain of infectious disease. It includes all words that are acceptable values for a parameter, common *synonyms* of these words, and words used elsewhere by the system in describing the parameter (e.g., when translating a rule into English or requesting the value of the parameter).

2.4.1 Reducing the Question to Terminal Words

Each word in the dictionary has a synonym pointer to its terminal word (terminal words point to themselves). For the purpose of analyzing the question, a non-terminal word is considered to be equivalent to its (terminal) synonym.

Terminal words may have properties indicating:

1) that this word is an acceptable value for some clinical parameter(s)

2) that this word *always* implicates a certain clinical parameter, system list, or table (e.g. the word "identity" always implicates the parameter IDENTITY, which means the identity of an organism)

3) that this word *might* implicate a certain parameter, system list, or table (e.g. the word "positive" might implicate the parameter NUMPOS, which means the number of positive cultures in a series)

4) that this word is part of a phrase which can be thought of as a single word (examples of such phrases are "transtracheal aspiration", "how long", and "not sterile".

Table 1. Properties of Terminal Words

The first three properties are actually inverse pointers which are generated automatically from properties of the clinical parameters. Specifically, a word receives the "acceptable value" pointer to a parameter (property (1) above) if it appears in the parameter's list of acceptable values -- a list which is used during the consultation to check the user's response to a request for the parameter's value. Also, each clinical parameter, list, and table has an associated list of key words that are commonly used when talking about this parameter, list, or table. These words are divided according to how sure we can be that a doctor is referring to this parameter, list, or table when the particular word is used in a question. It is from this list that terminal words' "implication" pointers (properties 2 and 3 in Table 1) are generated.

During the first phase of parsing, each word in the original text is replaced by its terminal word. For words not found in the dictionary, the system uses Winograd's root-extraction algorithm [8] to see if the word's lexical root is in the dictionary (e.g., the root of "decision" is "decide"). If so, the word is replaced by the terminal word for its root. Words still unrecognized after root extraction are left unchanged.

The resulting list of terminal and unrecognized words is then passed to a function which recognizes phrases. Using property 4 (see Table 1) of the terminal words in this list, the function identifies a phrase and replaces it with a single synonymous terminal word (whose dictionary properties may be important in determining the meaning of the question).

2.4.2 Classifying the Question

The next step is to classify the question so that the program can tell which specialist should answer it. Since all questions about the consultation must be about some specific context, the system requires that the name of the context (e.g., ORGANISM-1) be stated explicitly. This gives an easy way to separate general questions about the knowledge base from questions about a particular consultation. Further classification is done through a pattern matching approach similar to that used by Colby [1].

The list of words created by the first phase is tested against a number of patterns (about 50 at present). Each pattern has a list of actions to be taken if the pattern is matched. These actions set flags which indicate what type of question was asked. In the case of questions about judgmental knowledge (called *rule-retrieval* questions), pattern matching also divides the question into the part referring to the rule's PREMISE and the part referring to its ACTION. For example, in "How do you decide that an organism is streptococcus?", there is no PREMISE part, and the ACTION part is "an organism is streptococcus"; in "Do you ever use the site of the culture to determine an organism's identity?", the PREMISE part is "the site of the culture" and the ACTION part is "an organism's identity".

2.4.3 Determining What Pieces of Knowledge are Relevant

The classification of a question guides its further analysis. Each question type has an associated template with blanks to be filled in from the question. The different blanks and the techniques for filling them in are listed in Table 2. With the question correctly classified, the general question-answerer can tell which specialist should answer it. Filling in all blanks in the template gives the specialist all the information needed to find the answer.

1) <cntxt> - The context must be mentioned by name.

2) <rule> - Either a rule's name (RULE047) will be mentioned, or the word "rule" will appear, together with the rule's number (47).

3) <value> - One of the terminal words in the question has a dictionary property indicating that it is a legal value for the parameter (property 1, Table 1 -- e.g., THROAT is a legal value for the parameter SITE).

4) <parm> - All of the words in the list are examined to see if they implicate any clinical parameters. Strong implications come from words with properties showing that the word is an acceptable value of the parameter, or that the word *always* implicates that parameter (properties 1 and 2, Table 1). Weak implications come from words with properties showing that they *might* implicate the parameter (property 3, Table 1). The system uses an empirical scoring mechanism for picking out only the most likely parameters.
Associated with certain parameters are words or patterns which *must* appear in the question in order for the distinguish related parameters which may be implicated by the same key words in the first pass. For example, the word "PMN" implicates parameters CSFPOLY (the percent of PMNs in the CSF) and PMN (the percent of PMNs in the complete blood count). These are distinguished by requiring that the word "CSF" be present in a question in order for CSFPOLY to be implicated.

5) <list> - System lists are indicated in a manner similar to parameters, except that scoring is not done. Lists, like parameters, may have associated patterns which must be present in the question. Furthermore, lists have properties telling which other system lists are their subsets. If a question implicates both a list and a subset of that list, the more general (larger) list is discarded. As an example, the question "Which drugs are aminoglycosides?" implicates two lists: The list of all drugs and the list of drugs which are aminoglycosides. The system only considers the more specific list of aminoglycosides when answering the question.

6) <table> - Tables are indicated in a manner similar to lists except that an entry in the table must also be present in the question. For example, the word "organism" may indicate two tables: one containing a classification of organisms, and the other containing normal flora of various portals. The question "What organisms are considered to be subtypes of Pseudomonas?" will correctly implicate the former table, and "What are the organisms likely to be found in the throat?" will implicate the latter, because PSEUDOMONAS is in the first table and THROAT is in the second.

Table 2. Mechanisms for Analyzing a Question

```
** WHEN DO YOU DECIDE THAT AN ORGANISM IS A CONTAMINANT?

[1] Terminal words:  WHEN DO YOU CONCLUDE THAT A ORGANISM
                     IS A CONTAMINANT

[2] Question type:    Rule retrieval
    Premise part:    (WHEN DO YOU CONCLUDE)
    Action part:     (THAT A ORGANISM IS A CONTAMINANT)

[3] vocab. clues:    (WHENINFECT (ANY) 1) (WHENSTOP (ANY) 1)
       (Premise)     (WHENSTART (ANY) 1) (DURATION (ANY) 1)
    vocab. clues:    (CONTAMINANT (ANY) 4) (FORM (ANY) 1)
       (Action)      (SAMEBUG (ANY) 1) (COVERFOR (ANY) 1)

[4] Final translation:
        Premise: ANY
        Action: (CONTAMINANT ANY)

[5] The rules listed below conclude about:
         whether the organism is a contaminant
    6, 31, 351, 39, 41, 42, 44, 347, 49, 106
    Which do you wish to see?
         ** 6

    RULE006
    -------
      If:  1) The culture was taken from a sterile source, and
           2) It is definite that the identity of the organism
              is one of: staphylococcus-coag-neg bacillus-
              subtilis corynebacterium-non-diphtheriae
      Then: There is strongly suggestive evidence (.8)
            that the organism is a contaminant
```

Figure 8. Sample of MYCIN's Analysis of a Question
[User input follows the double asterisks.]

[1] The question is reduced to a list of terminal words.

[2] Pattern matching classifies the question as a rule-retrieval question, and divides it into a premise part and an action part.

[3] Dictionary properties of the terminal words are used to determine which parameters (and their values) are relevant to each part of the question. These vocabulary clues are listed in the form (<parm> (<values>) weight) where weight is used by the scoring mechanism to determine which parameters *should be eliminated from consideration.*

[4] After selecting only the most strongly indicated parameters, the final translation tells what rules can answer the question: there are no restrictions on the PREMISE, and the ACTION must contain the parameter CONTAMINANT (with any value).

[5] The answer consists of finding all rules which meet these restrictions, and printing those that the user wants to see.

2.5 Answering the Question

Corresponding to each question type, there are a number of possible answer templates. For example, for questions of the form "How do you know the value of <parm> of <cntxt>?", two of the answer templates are:

```
I used <rule> to conclude that <parm> of <cntxt> is <value>.
        This gave a cumulative c.f. of <certainty factor>.
        The last question asked before the conclusion was made
        was <question number>.

In answer to question <question number> you said that <parm> of
        <cntxt> is <value>.
```

The specialist for answering questions of a given type will need to check the history tree or the system's knowledge base in order to determine which of the answer templates is appropriate for a particular question. Some blanks in the answer template will be filled in by the same items that filled blanks in the question template. The remainder are filled by the specialist with information which will answer the question. In the above example, the slots <parm>, <cntxt>, and possibly <value> would be filled in from the question, and the other slots would be filled from the history tree.

2.5.1 Questions about MYCIN's Static Knowledge

General questions about MYCIN's factual knowledge are the easiest to answer. The specialist that provides answers to such questions must look up the desired information in the system's static knowledge base. Generally, an answer consists of simply displaying this information. As Figure 9 illustrates, the most processing that is ever done is to check the retrieved information against some item in the question (for equality of set membership).

The specialist that answers questions about judgmental knowledge is slightly more complicated. Answering these questions (Figure 10) involves built-in knowledge about the rule set, plus the ability to "read" the rules. By the time the question has been analyzed, the specialist knows exactly which parameters must appear in the PREMISE, and which must appear in the ACTION of any rule which answers the question. Values may be specified for any of the parameters. To answer the question, the rule-retrieval specialist must first find every rule whose PREMISE and ACTION satisfy these constraints. To do this, it needs to know that there are two special lists associated with each parameter: one containing every rule that uses the parameter in its PREMISE, and the other containing every rule that concludes about the parameter in its ACTION. Using these lists for the various parameters mentioned in the question, the specialist can find those rules that might answer the question. If no values were specified, the job is done and the relevant rules can be displayed without further analysis; otherwise, it is necessary to read each of the rules in the list and to eliminate those which do not mention the correct values for the parameter.

The rule-retrieval specialist also makes use of a piece of MYCIN's knowledge which was not discussed earlier. The system contains models of its own knowledge (called *rule models*) which are used primarily during acquisition of new medical knowledge from an expert [4]. These models, however, can be put to many uses -- one is to explain general patterns in decision making. The rule models are abstract descriptions of a subsets of rules and are generated automatically by reading the rules. For example, the model for IDENT-IS-PSEUDOMONAS tells what features are common to the majority of rules which conclude that the identity of an organism is pseudomonas.

If a model exists describing the rules about which the question is asking, the rule-retrieval specialist incorporates this model's information into its answer (Figure 11). Thus the question-answering facility is able to give some information about strategies for achieving some of the system's goals, as well as the individual rules which use the strategies.

```
** IS BLOOD A STERILE SITE?
Yes.  Blood is one of those sites that are normally sterile.

** WHAT ARE THE NONSTERILE SITES?
Those sites that are normally nonsterile are:  CERVIX, CUTANEOUS-ULCER,
     LOCHIA, NOSE, SKIN, STOOL, THROAT, URETHRA, VAGINA.

** WHAT ORGANISMS ARE LIKELY TO BE FOUND IN THE THROAT?
The likely pathogens associated with throat are:  STREPTOCOCCUS-
     PNEUMONIAE, STREPTOCOCCUS-GROUP-A, NEISSERIA-MENINGITIDIS.

** IS BACTEROIDES AEROBIC?
No, the aerobicity of bacteroides is anaerobic.

** WHAT DOSAGE OF STREPTOMYCIN DO YOU GENERALLY RECOMMEND?
The suggested dosage and route of streptomycin is 7.5 mg/kg q12h IM (or IV).

** HOW DO YOU TREAT MENINGOCOCCAL BACTEREMIA?
For treatment of NEISSERIA-MENINGITIDIS in PRIMARY-BACTEREMIA
Select therapy from among the following drugs (ranked according
   to relative efficacy, assuming uniform sensitivity):
   PENICILLIN        (.91)
   AMPICILLIN        (.9)   dose: 30 mg/kg q6h
   SULFISOXAZOLE     (.8)   dose: 60 mg/kg IV then 15 mg/kg q4h IV
                                  for 48 days then 15 mg/kg q4h PO
   CHLORAMPHENICOL   (.7)
   ERYTHROMYCIN      (.5)
   TETRACYCLINE      (.4)
Duration of therapy: 14 days
```

Figure 9. Questions about Factual Knowledge

```
** WHY DO YOU ASK WHETHER THE PATIENT HAS A FEVER OF UNKNOWN
   ORIGIN?
The rules listed below use:
     whether the patient has a true fever of unknown origin

RULE180
-------
  If:  1) The site of the culture is blood, and
       2) The number of cultures in the series including the
          culture is greater than or equal to 1, and
       3) The number of cultures in this series which were
          positive for the organism is less than or equal to 1,
          and
       4) The patient has a true fever of unknown origin, and
       5) Cardiac-surgery is a relevant item from the history of
          the patient
  Then: There is suggestive evidence (.6) that the infection is
        infective-endocarditis

** WHAT DOES GU MANIPULATION TELL YOU ABOUT THE IDENTITY OF AN
   ORGANISM?
The rules listed below use:
     whether the patient has had a genito-urinary manipulative
        procedure
to conclude about:
     the identity of the organism
156, 163, 190
Which do you wish to see?
     ** 156

RULE156
-------
  If:  1) The site of the culture is blood, and
       2) The gram stain of the organism is gramneg, and
       3) The morphology of the organism is rod, and
       4) The portal of entry of the organism is urine, and
       5) The patient has not had a genito-urinary manipulative
          procedure, and
       6) Cystitis is not a problem for which the patient has
          been treated
  Then: There is suggestive evidence (.6) that the identity of
          the organism is e.coli
```

Figure 10. Rule-Retrieval Questions

```
** HOW DO YOU DECIDE THAT AN ORGANISM MIGHT BE PSEUDOMONAS AERUGINOSA?
     Rules which conclude that the identity of the organism is
     pseudomonas-aeruginosa generally use one or more of the
     following pieces of information:
          the site of the culture
          the gram stain of the organism
          the morphology of the organism
     Furthermore, the following relationships hold:
     The gram stain of the organism, and the morphology of the
          organism tend to appear together in these rules.
     RULE184, RULE116, RULE047, RULE085, RULE040  conclude that
          the identity of the organism is pseudomonas-aeruginosa.
          Which  of these do you wish to see?
          ** 184
     RULE184
     -------
       If:  1) The category of the organism is not known, and
            2) The gram stain of the organism is gramneg, and
            3) The morphology of the organism is rod, and
            4) The aerobicity of the organism is facul
       Then:  There is weakly suggestive evidence (.1) that the
                 identity of the organism is pseudomonas-aeruginosa
```

Figure 11. Question Which Uses Rule Models

2.5.2 Consultation-Specific Questions

One of the simplest questions about a specific consultation inquires about the value of a particular parameter (Figure 12). The specialist which answers these questions must know how to retrieve this information. In some cases it will also be necessary to make comparisons to see whether a value specified in the question is one of the values which was deduced for that parameter.

```
** TO WHAT CLASS DOES ORGANISM-1 BELONG?
The category of ORGANISM-1 is enterobacteriaceae (.9)

** IS ORGANISM-1 A STREPTOCOCCUS?
No.
The identity of ORGANISM-1 is e.coli (.74) bacteroides (.7)
     klebsiella-pneumoniae (.4) pseudomonas-aeruginosa (.22)
     enterobacter (.21) proteus-non-mirabilis (.19) serratia (.03)
     salmonella (.02)
```

Figure 12. Questions about a Parameter's Value

Answering other types of consultation-specific questions involves the use of knowledge about the control structure, including:

1) how the system acquires information
2) why the system tries to find out about parameters
3) how information is used once it is acquired
4) what causes a rule to be tried
5) what causes a rule to fail.

The specialist for answering questions like "How do you know the value of <parm> of <cntxt>?", knows that the value of a parameter can come from two sources: it can be deduced by rules, or the user can tell us about it in response to a question. The history tree (see Section 2.2) will show which (possibly both) of these sources provided the particular information mentioned in the question (Figure 13).

```
** HOW DO YOU KNOW THAT CULTURE-1 WAS FROM A STERILE SOURCE?
I used rule189 to conclude that this blood culture was taken from
     a sterile source. This gave a cumulative c.f. of (1.0). The
     last question asked before the conclusion was made was 7.
** DID YOU CONSIDER BACTEROIDES AS A POSSIBILITY FOR ORGANISM-1?
Yes.  I used rule095 to conclude that the identity of ORGANISM-1
     is bacteroides. This gave a cumulative c.f. of (.7). The
     last question asked before the conclusion was made was 20.

** HOW DO YOU KNOW THAT ORGANISM-1 IS E.COLI?
The following rules made conclusions about whether the identity
     of ORGANISM-1 is e.coli

          cumulative certainty       last question asked
 RULE          Yes      No       before conclusion was made
RULE021        (.47)                      20
RULE084        (.55)                      22
RULE003        (.74)                      24
In answer to question 9 you said that the identity of ORGANISM-1
   is e.coli (.3)
```

Figure 13. Question Regarding How a Conclusion Was Made

If the question is phrased in the negative, it is necessary first to find all the ways the conclusion could have been made (this is a simple task of rule-retrieval), then to explain why it *wasn't* made in this consultation (Figure 14). The specialist for answering these questions must know what situations can prevent conclusions from being made. The second question in Figure 14 illustrates how the answer to one question might cause another question to be asked.

```
** WHY DID YOU RULE OUT STREPTOCOCCUS AS A POSSIBILITY FOR
   ORGANISM-1?
The following rules could have been used to determine that the
     identity of ORGANISM-1 is streptococcus:  RULE033.  However,
     none of these succeeded in the context of ORGANISM-1.
     If you would like an explanation for why any of these
     rules failed, please enter their numbers:
     ** 33
Clause 2 of rule033 ["the morphology of the organism is coccus"]
     was already known to be false for ORGANISM-1, so the rule
     was never tried.

** WHY DON'T YOU THINK THAT THE MORPHOLOGY OF ORGANISM-1 IS
   COCCUS?
It is definite that the morphology of ORGANISM-1 is rod. Knowing
     this with certainty rules out all other values for the
     the morphology of ORGANISM-1, including coccus.
```

Figure 14. Questions Regarding Why a Conclusion Wasn't Made

The specialist for answering questions of the form "How did you use <parm> of <cntxt>?", needs to know not only how to find the specific rules which might use a parameter, but also how a parameter can cause a rule to fail and how one parameter can prevent another from being used. The history tree can be checked to see which of the relevant rules used the parameter, which failed because of the parameter, and which failed for some other reason, preventing the parameter from being used (Figure 15).

```
** HOW DID YOU USE THE AEROBICITY OF ORGANISM-1?
The aerobicity of ORGANISM-1 was used in the following rules:
     RULE003, RULE005, RULE084.
The aerobicity of ORGANISM-1 caused the following rules to fail:
     RULE035, RULE051, RULE052, RULE053, RULE111.
The aerobicity of ORGANISM-1 also would have been used in:
     RULE037, RULE050, RULE058, RULE086, RULE110, RULE184,
     RULE203, RULE204, RULE205.  However, none of these
     succeeded in the context of ORGANISM-1.  If you would
     like an explanation for why any of these rules failed,
     please enter their numbers:
     ** 37
Rule037 was tried in the context of ORGANISM-1, but it failed
     due to clause 1 ["the category of the organism is not
     known"].

** DID YOU CONSIDER THE FACT THAT PATIENT-1 IS NOT A COMPROMISED
   HOST?
The fact that Fred Jones is not a compromised host caused the
     following rules to fail:  RULE085, RULE106.
The fact that Fred Jones is not a compromised host also would
     have been used in:  RULE109.  However, none of these
     succeeded in the context of PATIENT-1.  If you would
     like an explanation for why any of these rules failed,
     please enter their numbers:
     ** NONE
```

Figure 15. Questions Regarding How Information Was Used

For questions of the form "Why didn't you find out about <parm> of <cntxt>?", general knowledge of MYCIN's control structure tells the conditions under which it would have been necessary to find out some piece of information. The record of the consultation can be used to determine why these conditions never arose for the particular parameter in question (Figure 16).

Figure 16 also illustrates that MYCIN's GQA allows a user to get as much information as is desired. The first answer given was not really complete in itself, but it led the user to ask another question to get more information. Then another question was asked to determine why clause 1 of RULE159 was false. The answers to the first two questions both mentioned rules which could be printed if the user wanted to examine them. The special command PR (Print Rule) is for the user's convenience. It requires no natural-language processing and thus can be understood and answered immediately. ("What is rule 109?" or "Print rule 109." also would have been understood.)

```
** DID YOU FIND OUT ABOUT THE CBC ASSOCIATED WITH CULTURE-1?
During the preceding consultation, I never had reason to try to find out
     whether the results of the CBC from the time period of this blood
     culture are available.  It would have been necessary to know this
     information in order to use any of the following: RULE109.  However,
     none of these succeeded in the context of this blood culture.  If you
     would like an explanation for why any of these rules failed, please
     enter their numbers:
     ** 109

Rule109 would be relevant to this blood culture when applied to: ORGANISM-1.
I never had reason to try to apply rule109 to ORGANISM-1.  I would have
     needed to use rule109 in order to find out whether ORGANISM-1 is a
     contaminant.  However, during the preceding consultation, I never had
     reason to try to find out whether ORGANISM-1 is a contaminant.


** WHY DIDN'T YOU NEED TO KNOW WHETHER ORGANISM-1 IS A CONTAMINANT?
During the preceding consultation, I never had reason to try to find out
     whether ORGANISM-1 is a contaminant.  It would have been necessary to
     know this information in order to use any of the following:  RULE006,
     RULE007, RULE106, RULE108, RULE109, RULE159. However, none of these
     succeeded in the context of ORGANISM-1.  If you would like an
     explanation for why any of these rules failed, please enter their
     numbers:
     ** 159

Rule159 was tried in the context of ORGANISM-1, but it failed due to clause
     1 ["it is suspected that the identity of the organism is
     corynebacterium-non-diphtheriae"]


** IS ORGANISM-1 CORYNEBACTERIUM-NON-DIPHTHERIAE?
     No.  The identity of ORGANISM-1 is e.coli (.74) bacteroides (.7)
        klebsiella-pneumoniae (.4) pseudomonas-aeruginosa (.22) enterobacter
        (.21) proteus-non-mirabilis (.19) serratia (.03) salmonella (.02)


** PR 109
RULE109
-------
     If:  1) The culture was taken from a sterile source, and
          2) It is not known whether the organism is a contaminant, and
          3) The patient is not a compromised host, and
          4) The patient was febrile due to the infection, and
          5) The results of the cbc from the time period of the culture are
             available, and
          6) A - The white count from the CBC is greater than 10.5, or
             B - The percent of pmn's in the cbc at the time the culture was
                obtained is greater than 78, or
             C - The percent of wbc's which were bands in the CBC at the
                time the culture was obtained is greater than 10
     Then:  There is strongly suggestive evidence (.8) that the organism is
            not a contaminant
```

Figure 16. Questions Regarding Why a Parameter Wasn't Traced

In questions asking about the application of a rule to a context there are three possibilities: the rule told us something about the context; the rule failed when applied to that context; or the rule was never tried in that context. The history tree tells which of these is the case. Furthermore, if a rule succeeded, there is a record of all the conclusions it made, and if it failed, the reason for failure is recorded. As Figure 17 illustrates, rules are only applicable in certain contexts. If the rule could not apply to the context mentioned in the question, but might be relevant when applied to some other context [which is an ancestor or descendent in the context tree], then the question is answered as if it had mentioned the relevant context.

There are interactions among the different types of questions which weren't mentioned earlier for the sake of clarity. Sometimes a question may indicate an incorrect assumption about the state of MYCIN's knowledge. For example a question could ask "Why don't you think that ORGANISM-1 is E.coli?" when, in fact, the system has concluded that the organism *is* E.coli. To answer this question, the explanation system would explain how it *did* make such a decision, despite the phrasing of the question. Similarly, when asked how the system used (or concluded about) some parameter that was never traced, the system explains why it didn't need to find out about the parameter, regardless of how the question was phrased.

```
** DID RULE 116 TELL YOU ANYTHING ABOUT INFECTION-1?
Rule116 would be relevant to the primary-bacteremia when applied
     to: ORGANISM-1.
Clause 5 of rule116 ["the infection is meningitis"] was already
     known to be false for ORGANISM-1, so the rule was never tried.

** HOW WAS RULE 178 HELPFUL WHEN YOU WERE CONSIDERING ORGANISM-1?
Rule178 would be relevant to ORGANISM-1 when applied to:
     CULTURE-1.
I never had reason to try to apply rule178 to this blood culture.
     I would have needed to use rule178 in order to find out
     whether this blood culture was taken from a sterile source.
     However, I was already certain that this blood culture was
     taken from a sterile source.

** WHAT DID RULE 295 TELL YOU ABOUT ORGANISM-1?
I used rule295 to conclude that the identity of ORGANISM-1 is
     hemophilus-influenzae. This gave a cumulative c.f. of (.25).
     The last question asked before the conclusion was made
     was 36.

** WHY DIDN'T YOU USE RULE112 TO FIND OUT ABOUT ORGANISM-1?
RULE112 was not executed because it would have caused circular
     reasoning when applied to ORGANISM-1.  Would you like to
     see the chain of rules and parameters which makes up this
     circle?
     ** YES
I wanted to know about the identity of ORGANISM-1 because I
     try to find out the identity of the organism for all
     current organisms of the patient.
To find out about the identity of ORGANISM-1, I tried to use
     rule021. Before I could use rule021, I needed to know about
     a prior organism with possibly the same identity as
     ORGANISM-1.
To find out about a prior organism with possibly the same
     identity as ORGANISM-1, I tried to use rule005. Before I
     could use rule005, I needed to know about the aerobicity of
     ORGANISM-1.
To find out about the aerobicity of ORGANISM-1, I tried to use
     rule031. Before I could use rule031, I needed to know about
     the category of ORGANISM-1.
To find out about the category of ORGANISM-1, I tried to use
     rule112. Before I could use rule112, I needed to know about
     the identity of ORGANISM-1.
But this is the unknown parameter I sought originally.
```

Figure 17. Question Regarding the Application of a Rule

3 Conclusions

Consultation systems which give expert advice in some domain form one class of artificial intelligence programs which can provide useful solutions to real-world problems. The utility of such a system, however, depends on its acceptability to human users. One feature which can increase a system's acceptability is a mechanism whereby the system can explain or justify its advice.

The development of an explanation mechanism for a consultation system is very much related to the problems of representing knowledge and of making use of different sources of knowledge. Since the production system formalism provides a unified way to represent modular pieces of knowledge, the task of designing an explanation capability is simplified for production-based consultation systems. The example of MYCIN shows how this can be done and illustrates further that a system designed for a single domain with a small, technical vocabulary can give comprehensive answers to a wide range of questions without sophisticated natural-language processing.

Acknowledgments

The authors wish to express their gratitude for the interest and advice of Drs. Bruce Buchanan and Cordell Green (Computer Science Department). We also are indebted to the following MYCIN Project co-workers: Jan Aikins, Stanton Axline, Stanley Cohen, Larry Fagan, Frank Rhame, Bill van Melle, Sharon Wraith, and Victor Yu. Special thanks are due to Bill van Melle and Bruce Buchanan who made numerous helpful comments on earlier drafts of the paper.

References

[1] Colby, K.M., Parkison, R.C., and Faught, B. Pattern-Matching Rules for the Recognition of Natural Language Dialogue Expressions. AI Memo 234, Stanford Artificial Intelligence Laboratory, Stanford University, June 1974.

[2] Davis, R., and King, J. An Overview of Production Systems. To appear in *Machine Representations of Knowledge*, published as *Machine Intelligence 8* (eds. E.W. Elcock and D. Michie), John Wylie, December 1976. Also available as AI Memo 271, Stanford Artificial Intelligence Laboratory, Stanford University, October 1975.

[3] Davis, R., Buchanan, B.G., and Shortliffe, E.H. Production Rules as an Approach to Knowledge-Based Consultation Systems. AI Memo 266, Stanford Artificial Intelligence Laboratory, Stanford University, October 1975. Also accepted for publication in *Artificial Intelligence*, February 1977.

[4] Davis, R. Applications of Meta Level Knowledge to the Construction, Maintenance and Use of Large Knowledge Bases. Doctoral dissertation, Stanford University, June 1976. Also available as AI Memo 283, Stanford Artificial Intelligence Laboratory, Stanford University, July 1976.

[5] Shortliffe, E.H., Axline, S.G., Buchanan, B.G., Merigan, T.C., Cohen, S.N. An Artificial Intelligence Program to Advise Physicians Regarding Antimicrobial Therapy. *Computers and Biomedical Research*. 6,544-560, 1973.

[6] Shortliffe, E.H., Davis, R., Buchanan, B.G., Axline, S.G., Green, C.C., Cohen, S.N. Computer-Based Consultations in Clinical Therapeutics: Explanation and Rule-acquisition Capabilities of the MYCIN System. *Computers and Biomedical Research* 8,303-320, 1975.

[7] Shortliffe, E.H. MYCIN: A Rule-Based Computer Program to Advise Physicians Regarding Antimicrobial Therapy Selection. Doctoral dissertation, Stanford University, October 1974. Also available as *Computer-Based Medical Consultations: MYCIN*, American Elsevier, 1976.

[8] Winograd, T. Undersanding Naural Language. *Cognitive Psychology* 3,1-191, 1972.