Report 78-26
Stanford -- KSL

Scientific DataLink

An Antibiotic Therapy Selector which Provides for Explanations.
William J. Clancey,
Dec 1978

card 1 of 1

Stanford Heuristic Programming Project
HPP-78-26

December 1978

AN ANTIBIOTIC THERAPY SELECTOR WHICH PROVIDES FOR EXPLANATIONS

by

William J. Clancey

COMPUTER SCIENCE DEPARTMENT
Stanford University
Stanford, California 94305

AN ANTIBIOTIC THERAPY SELECTOR WHICH PROVIDES FOR EXPLANATIONS

by

William J. Clancey

Heuristic Programming Project
Computer Science Department, Stanford University
Stanford, California 94305, U.S.A

HPP 78-26 December 1978

Abstract

A program for selecting antibiotic therapy is described that makes sophisticated recommendations and provides simple, useful explanations of its reasoning for a user. Our method is to structure the therapy optimization problem in terms of local and global solution criteria that are applied in a generate-and-test algorithm. Generation of therapy recommendations is directed by a fixed, ordered set of canonical instructions that describe the global characteristics of a recommendation. Other factors are dealt with in a "planning" phase particular to each organism for which therapy is to be prescribed, and the test phase that incorporates patient considerations such as allergies and age. We demonstrate the advantages of this canonical form for explanation, comparison of alternative recommendations, and a simple instructive capability.

Table of Contents

List of Figures

1 Introduction

A program that is designed to provide sophisticated expert advice must cope with the needs of naive users who may find the advice puzzling or hard to accept if they don't understand it. This paper describes a program that provides for explanations of its decisions. Its problem domain is antibiotic therapy selection, an optimization problem that seeks to provide "coverage" for organisms (causing an infectious disease), while minimizing the number of drugs prescribed. There are many factors to consider, such as prior therapies and drug sensitivities, and a human often finds it hard to juggle all of the constraints at once. When the "optimal" solution is provided by a computer program, its correctness may not be immediately obvious to many users, and this motivates our desire to provide an explanation capability to justify the program's results.

The explanation capability of our program derives from two basic programming considerations. First, we have used heuristics ("rules of thumb") that capture what expert physicians consider to be good medical practice. Thus, while the program is not designed to mimic the step-by-step problem-solving behavior of a physician, its chief decision criteria have been provided by expert physicians and so it is at least a plausible assumption that they will be comprehensible to other physician users.

The second programming consideration is that the program leaves behind traces of decisions that were made. These are used for explaining what occurred during the optimization process and why the output was not different. While leaving behind traces for an explanation routine is not new (e.g, see [11], [1], [8], [9]), the means that we use to retrieve them is novel, namely a state transition diagram representation of the algorithm. Our work

demonstrates that a cleanly structured algorithm can provide both sophisticated performance and a simple, useful explanation capability.

2 The MYCIN Task

In a MYCIN consultation [10] a user at a computer terminal answers questions about a patient suspected to have an infectious disease. Typical factors he supplies are laboratory tests and the patient's living and working conditions. The user receives advice about organisms to treat and therapy to cover them (Figure 1).

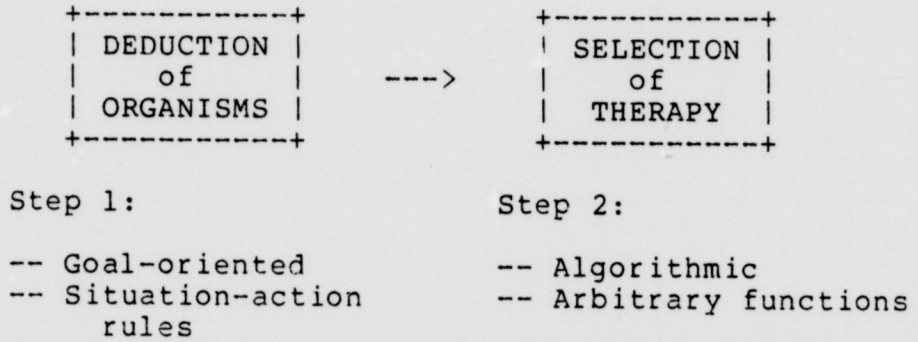

Figure 1. Two Basic Steps in a MYCIN Consultation

The first step, deduction of organisms, is goal-oriented. Situation-action rules [10] [5] are used by backward chaining to focus the process of gathering data from the user in order to reach conclusions about the patient. In the second step, selection of therapy, there are many interacting factors and iterative operations. This algorithmic knowledge is represented by INTERLISP functions. We will be concerned with here with the therapy selector (1).

---

(1) References [10] and [7] describe the explanation capability for the rule-based step.

Figure 2 shows the result of a MYCIN consultation. The organisms to be treated, termed "items," are listed followed by the preferred therapy recommendation. Notice that two drugs were sufficient to cover for four organisms.

```
It is important to cover for the following probable
infection(s) and associated organism(s):

INFECTION-1 is MENINGITIS

     ITEM-1:  STAPHYLOCOCCUS-COAG-POS
     ITEM-2:  E.COLI
     ITEM-3:  KLEBSIELLA-PNEUMONIAE
     ITEM-4:  PSEUDOMONAS-AERUGINOSA


[REC-1] My preferred therapy recommendation is as follows:
   In order to cover for Items <1 2 3 4>:
        Give the following in combination:

        1) CHLORAMPHENICOL
        2) GENTAMICIN


Do you wish to see the next choice therapy?
** NO
```

Figure 2. Organisms to be Treated and the Recommendation

3 The Problem

We will now briefly describe the historical development of the therapy algorithm, leading up to the framework we currently use to derive both optimal therapy and simple explanations.

The main problem of the therapy selector is to prescribe the best drug for each organism thought to be causing the infection, while minimizing the total number of drugs. These two constraints often conflict: The best prescription for, say, four items may require four different drugs, although

in general no more than two drugs should be given (for reasons of drug interaction, toxic side effects, cost, etc.).

The original therapy program [10] lacked a general scheme for relating the local constraints (best drug for each item) to the global constraint (fewest possible number of drugs). As we began to investigate the complexities of therapy selection, it became necessary to patch the program to deal with the special cases we encountered. Before long we were losing track of how any given change would affect the program's output. We found it increasingly difficult to keep records during the program execution for later use in the explanation system; indeed, the logic of the program was too confusing to explain easily.

We decided to start over, aiming for a more structured algorithm that would provide sophisticated therapy, and by its very organization would provide simple explanations for a naive user. The question was: What organization could balance these two, sometimes contradictory, goals?

Because we wanted to formulate judgments that could be provided by physicians and would appear familiar to them, we decided not to use mathematical methods such as evaluation polynomials or Bayesian analysis [6]. On the other hand, MYCIN's inferential rule representation was clearly inadequate because of the general algorithmic nature of the problem (i.e., iteration and complex data structures). We turned our attention to separating out the optimization criteria of therapy selection from control information (specifications for iteratively applying the heuristics). As we discuss below, the key improvement was to canonically encode the optimization performed by the inner loop of the algorithm.

4 Our Solution

4.1 Local and Global Criteria

We found that viewing the optimization problem in terms of local and global criteria provided a fruitful means of structuring the problem. Local criteria are the item-specific factors, such as sensitivity of the organism to preferred therapies, toxicity of drugs, the desire to "save" drugs for more serious diseases, and the desire to continue current therapy if possible. Global criteria deal with the entire recommendation: We wished to minimize the number of drugs, prescribing only two drugs if possible to cover for all of the most likely organisms (2). In addition, there were a few patient factors to consider, such as allergies to antibiotics.

Besides providing for optimal therapy, we wished to provide for an explanation capability that would list simple descriptions of the therapy selection heuristics used by the algorithm, as well as reasons for not making a different prescription.

After clearly stating these design goals, we needed an implementation scheme that would bring about the optimization. The key to our solution was the use of a generate and test control structure for separately applying the local and global factors. Figure 3 shows the steps of the generate and test method, and below them, the corresponding steps of our algorithm. Briefly, the steps are to: 1) plan by ranking the drugs--the local factors are considered here; 2) propose a recommendation and test it, thus dealing with the global factors; and 3) prescribe a recommendation. The following sections consider these steps in more detail.

---

(2) Here we realized that we could group the items by those which definitely should be treated ("most likely") and those which could be left out when three or more drugs would be necessary.

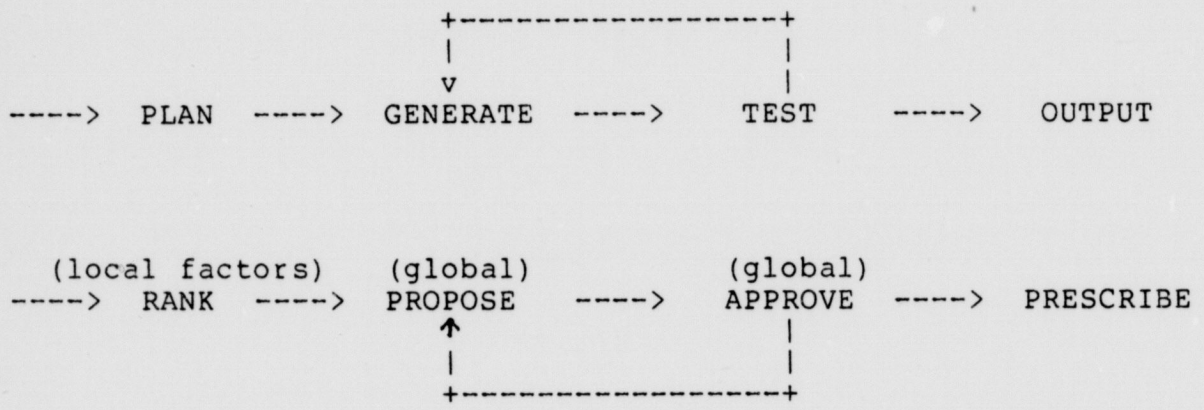

Figure 3. Therapy Selection Viewed as a Generate and Test Process

4.2 Plan

We start with an a priori list of drugs for each organism and sort it by applying situation-action rules for ranking. These reranking rules are applied independently for every organism to be treated. The chief purpose of this sorting process is to incorporate drug sensitivity information for the organisms growing on cultures taken from the patient. (A typical rule might be, "if the organism growing on the culture appeared to be insensitive to the drug, then classify the drug as third choice.") Thus, we arrive at a patient-specific list of drugs for each organism, reranked and grouped into first, second, and third choice.

Because this sorting process is a consideration specific to each organism, we refer to it as a local criterion of optimal therapy. We call it (loosely) a "planning step" because it makes preparations for later steps.

4.3 Generate

The second step of the algorithm is to take the ordered drug lists and

generate possible recommendations. This is done by a **proposer** which selects subsets of drugs (a recommendation) from the collection of drugs for all of the organisms to be treated. (Recall that the sorted organism-specific drug lists are partitioned; thus the collection is partitioned into first, second and third choice drugs.) Selection is directed by a fixed, ordered set of **instructions** that specify how many drugs to select from each preference group. The first few instructions are listed in Figure 4.

	Number of drugs of this rank:		
Instruction	"first"	"second"	"third"
1	1	0	0
2	2	0	0
3	1	1	0
4	1	0	1
.			
.			
.			

Figure 4. Instructions for the Therapy Proposer

For example, the third instruction tells the proposer to select a first choice drug and a second choice drug. Instructions for one and two-drug recommendations are taken from a static list; those for recommendations containing three or more drugs are generated from a simple pattern [3].

It should be clear that the ordering of the instructions ensures that two of the global criteria will be satisfied: prescribing one or two drugs if possible, and selecting the best possible drug(s) for each organism. An instruction therefore serves as a **canonical description** of a recommendation.

---

[3] The instructions of a single third choice drug and two third choice drugs are not valid, so it is not convenient to generate all instructions from one algorithm. In addition, there are a few special cases to deal with, such as combination drugs which are sometimes considered to play the role of a single drug for purposes of minimization. We will not deal with these exceptions here.

Consequently, we can "reduce" alternate subsets of drugs to this form (the number of drugs of each rank) and compare them. We will see some uses for this capability later (Section 6 and Section 7).

4.4 Test

Since all the drugs for all of the organisms were grouped together for use by the proposer, it is quite possible that a proposed recommendation will not cover for all of the most likely organisms. For example, the proposal might have two drugs which are first choice for one item, but are second or third for other items, or not even on their lists. Thus, the first step of testing is to make sure that all of the most likely items are covered.

The second test ensures that each drug is in a unique drug class. For example, a proposal having both Gentamicin and Streptomycin would be rejected because these two drugs are aminoglycosides and so cause a "redundant" effect.

The last test is for patient-specific contraindications. These rules take into account allergies, age of the patient, pregnancy, etc. (These rules are relatively expensive to apply, so they are done last, rather than applying them to each possible drug in the plan step.) With this test, we have dealt with the last global criterion of therapy selection.

The first proposal that satisfies these three tests is prescribed. Drug prescription won't be considered further here; it consists primarily of algorithmic dosage calculation and adjustment in the case of renal failure.

4.5 Performance

We have found that the algorithm described above is manageable and performs well. It is straightforward to add new rules for ranking the drugs

and for testing the proposals. The canonical instructions are relatively fixed, but it would not be difficult, for example, to provide infection-specific instruction sets. The program has made sophisticated recommendations for a library of over 100 meningitis patients. In a formal evaluation of the MYCIN program using this therapy selection routine, the program performed as well as faculty at the Stanford School of Medicine (physicians who did not take part in the program's development) [12].

5 The Explanation Capability

We will now consider how the structure of the algorithm is exploited to produce simple explanations. A sample user question about therapy selection is shown in Figure 5. The medical decisions that were applied to the drug Chloramphenicol are listed as a logical sequence of reasons that is produced by retrieving and printing traces which were left behind by the program. (The trace retrieval program is termed "CHRONICLER" because its explanations consist of a chronicle of decision events.)

```
** WHY DID YOU GIVE CHLORAMPHENICOL FOR E.COLI IN REC-1?

CHLORAMPHENICOL was prescribed for ITEM-2 in RECOMMENDATION-1:

Since
    -- CHLORAMPHENICOL is a treatment of choice for e.coli in
       meningitis
    -- ITEM-2 is sensitive to CHLORAMPHENICOL
    -- there were no contraindications for it

CHLORAMPHENICOL was prescribed because it was part of the
    recommendation that covers for all of the items, using the
    fewest number of drugs.
```

Figure 5. A Question Concerning Why a Drug was Prescribed

(User input follows the double asterisks.)

Figure 6 shows the general organization of the explanation system. The traces (discussed below) constitute a dynamic event history. A chronicle of events is printed by using a process transition diagram to selectively retrieve the relevant traces.

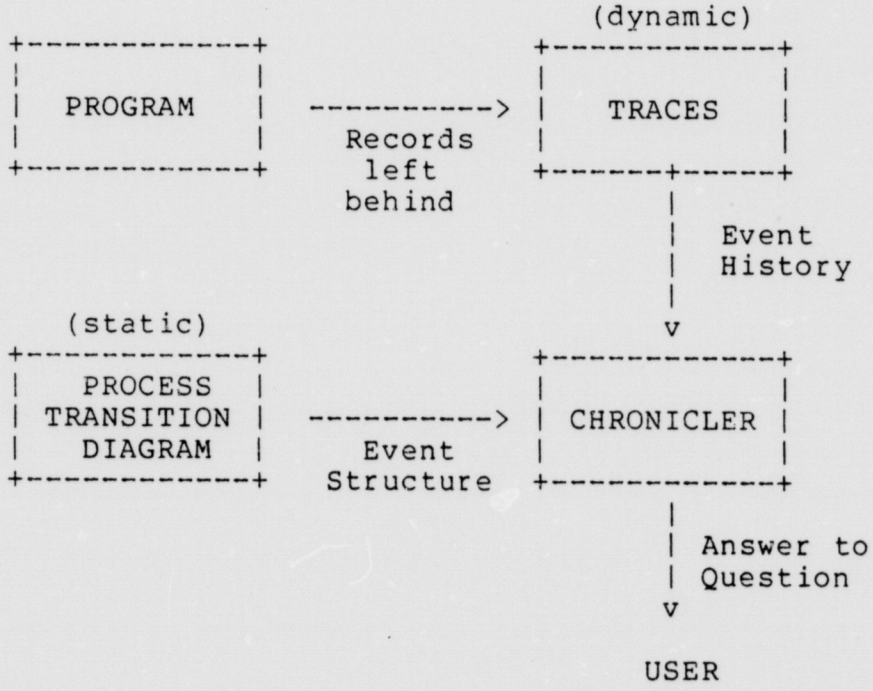

Figure 6. Organization of the Explanation System

Figure 7 shows the kind of transition diagram we use to represent the steps of therapy selection. The states roughly correspond to the generate and test steps shown in Figure 3. The arcs are labeled as positive and negative criteria; these correspond to the medical strategies, e.g., "the drug is on the treatment of choice list for the organism (the a priori list) and so was considered to cover for the organism." If a drug is prescribed, there must be a sequence of positive criteria leading from the first state to the output state. These are the reasons we offer the user as an explanation for prescribing the drug. To make the explanation clearer, the states are reordered to conform to the general scheme:

"Since
-- <plan criteria>
-- <test criteria>

(therefore)
<generate and output criteria>"

On the other hand, if a drug is not prescribed, there must be a negative criterion to explain why it dropped out of contention. Failure to prescribe can be caused by either failure to consider the drug (plan) or failure of a test. A third possibility is that the drug wasn't part of an acceptable recommendation, but was otherwise a plausible choice (when considered alone). In this case, the drug needs to be considered in the context of a full recommendation for the patient. See Figure 11 for an example [4].

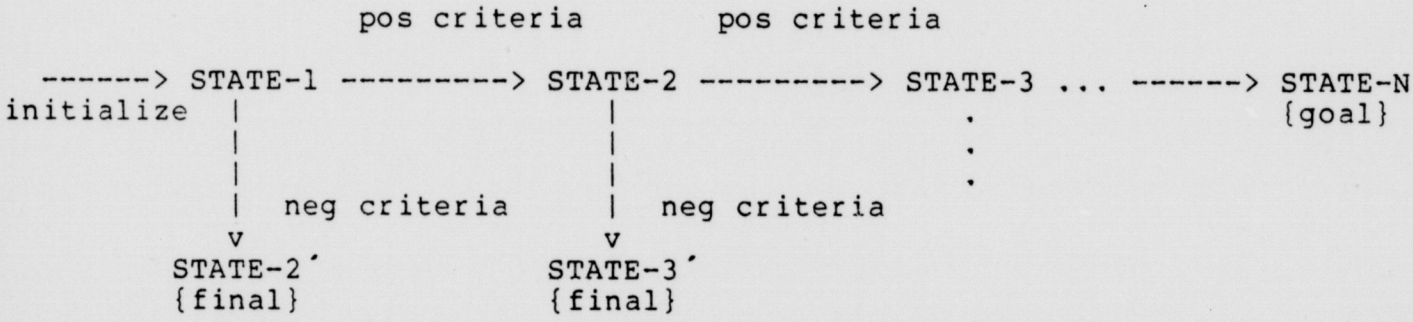

Figure 7. The Process Transition Diagram

---

[4] Events are recorded as properties of the drugs they involve. The trace includes other context such as the item being considered. To deal with iteration, events are of two types: "enduring" and "pass-specific." Enduring events represent decisions that, once made, are never reconsidered, e.g., the initial ranking of drugs for each organism. Pass-specific events may not figure in the the final result; they may indicate computation that failed to produce a solution, e.g., proposing a drug as part of a specific recommendation. Thus, traces are accessed by drug name and the context of the computation, including which pass of the generate and test process produced the final solution.

Figure 8 shows an example of a question concerning why a drug wasn't prescribed. In response to a question of this type, the negative criterion is printed and the user is offered an opportunity to see the positive decisions accrued up this point.

```
** WHY DIDN'T YOU SUGGEST PENICILLIN IN REC-1 FOR STAPH-COAG+ ?

PENICILLIN was not prescribed for ITEM-1 in RECOMMENDATION-1:

    PENICILLIN was discounted for ITEM-1 because it is NOT DEFINITE
    that the item is sensitive to this drug.
    There are other potential therapies under consideration which
    are much more desirable, viz, current therapies or drugs to
    which the item is definitely sensitive.

Would you like to see some details? ** YES

    The drugs to which the staphylococcus-coag-pos is sensitive are:
    cephalothin (1.Ø) vancomycin (1.Ø) gentamicin (1.Ø) tobramycin (1.Ø)
    erythromycin-and-tetracycline (1.Ø) chloramphenicol-and-erythromycin
    (1.Ø) [RULEØ98 RULE445]

Would you like to know about the history of PENICILLIN in the
decision process up to this point? ** YES

    -- PENICILLIN is a treatment of choice for staphylococcus-coag-
    pos in meningitis
    But as explained above, PENICILLIN was discounted.
```

Figure 8. Question Concerning Why a Drug was not Prescribed

In this example, we see that penicillin was not prescribed because it is not definite that the item is sensitive to this drug. That's the negative criterion. The fact that penicillin was a treatment of choice permitted its transition to the reranking step. This is shown in Figure 9. When MYCIN rules (not INTERLISP code) are used to make a transition decision, we can provide further details, as shown in Figure 8.

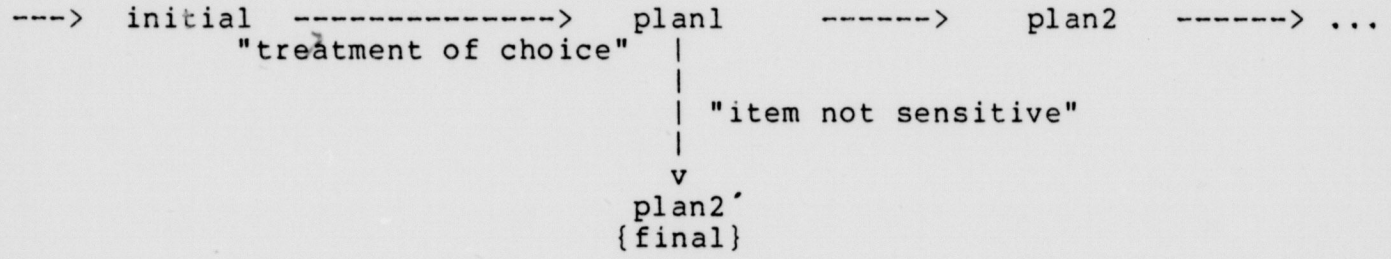

Figure 9. Trace History For the Question Shown in Figure 8

For questions involving two drugs, e.g., "Why did you prescribe Chloramphenicol instead of Penicillin for Item-1?" CHRONICLER is invoked to explain why the rejected drug wasn't given. Then the user is offered the opportunity to see why the other drug was given.

To summarize, we arrange in the program to leave behind traces that record the application of the positive and negative criteria. The explanation system uses a state transition diagram that represents the steps of the algorithm to retrieve the relevant traces in a logical order.

Finally, it is interesting to note that CHRONICLER is described well by Bobrow and Brown's Synthesis, Contingent Knowledge, and Analysis (SCA) paradigm for understanding systems [1]. "Contingent knowledge" is a record of program-synthesized observations for later use by an analysis program to answer questions or comment upon the observed system. In CHRONICLER, the traces and transition diagram constitute the contingent knowledge structure. Synthesis (abstraction of results) is performed by the therapy selector as it classifies the drugs in the various decision steps and records its "observations" in traces [5]. Analysis is performed by CHRONICLER as it

---

[5] The process of recording traces sometimes requires computation beyond that needed to solve the problem. For example, when one piece of code treats several cases the same, we may wish to record which case occurred.

"reads" the traces, interpreting them in terms of the process transition diagram. The meteorology question-answering system described by Brown [2] uses a similar knowledge representation.

6 Comparing Alternative Recommendations

The therapy program will print alternative recommendations should the user request. This is done by eliminating drugs from contention and rerunning the algorithm. Figure 10 shows an example of an alternative recommendation and a comparison of this alternative to the algorithm's first choice. The comparison is made by simply reducing the recommendations to canonical form and then determining their relative ranking in the list of instructions. A recommendation is better than another when it reduces to a better instruction, i.e., the preference description of its drugs appears earlier in the fixed list of instructions. It is also possible that the comparison will be a "close call," because one of the recommendations might use better drugs for the most likely organisms, but cover for fewer of the less likely organisms. Again, it is the ability to **encode output canonically** that gives us the ability to make such a direct comparison of alternatives.

7 Evaluating a User's Choice of Therapy

The comparison described above is useful to a physician who prefers to give one of the alternative regimens and wishes to know if the program truly considers it to be optimal therapy. However, it is tedious to request for all possible alternatives to be printed, so we offer the user the opportunity to enter his own set of drugs for the organisms that require therapy (Figure 11).

```
Do you wish to see the next choice therapy?
** YES
Which, if any, of the above drugs do you wish
eliminated from consideration?
** CHLORAMPHENICOL


[REC-2] Next best therapy recommendation:
   In order to cover for Items <2 3 4>:
        Give: GENTAMICIN

   In order to cover for Item <1>:
        Give: METHICILLIN


RECOMMENDATION-2 is substantially inferior to RECOMMENDATION-1
(because it requires more drugs or drugs that are not as good).

Would you like to see details of this comparison?
** YES

Looking at the coverage of the 'most likely' items, we see that:

     RECOMMENDATION-1 consists of 1 first choice drug (CHLORAMPHENICOL
     for ITEM-2 and ITEM-3) and 1 second choice drug (GENTAMICIN for
     ITEM-3 and ITEM-1)

     RECOMMENDATION-2 consists of 1 first choice drug (METHICILLIN for
     ITEM-1) and 1 third choice drug (GENTAMICIN for ITEM-2 and ITEM-3)


Do you wish to see the next choice therapy?
** NO
```

Figure 10. Comparing Alternative Recommendations

```
** WHY DIDN'T YOU GIVE AMPICILLIN-AND-GENTAMICIN FOR E.COLI IN REC-1?

AMPICILLIN-AND-GENTAMICIN was not prescribed for ITEM-2 in RECOMMENDATION-1:

AMPICILLIN-AND-GENTAMICIN is a plausible choice for e.coli in meningitis, and
was not explicitly rejected for use against ITEM-2 in RECOMMENDATION-1.
However, the best therapy did not include AMPICILLIN-AND-GENTAMICIN.

If you would like to suggest therapy which includes AMPICILLIN-AND-GENTAMICIN,
your regimen will be compared to MYCIN's.
Would you like to do this? ** Yes

For each item in turn, enter the drug you would have prescribed in
RECOMMENDATION-1.
In order to minimize the number of drugs in your recommendation, you may not
want to prescribe therapy for every item.  Items which represent the most
likely organisms are indicated with a plus sign (+).

 + ITEM-1 -- the staphylococcus-coag-pos ** GENTAMICIN
 + ITEM-2 -- the e.coli ** AMPICILLIN GENTAMICIN
 + ITEM-3 -- the klebsiella-pneumoniae ** GENTAMICIN
   ITEM-4 -- the pseudomonas-aeruginosa **

[Checking for contraindications...]
[Considering AMPICILLIN-AND-GENTAMICIN for use against INFECTION-1...]
[No contraindications found...]

[Now comparing your prescription to MYCIN's...]

Perhaps you did not realize that one of the drugs you prescribed, GENTAMICIN,
will cover for ITEM-4, an item for which you did not prescribe therapy.  I
have changed your prescription accordingly.


 ORGANISMS          Your regimen                    MYCIN's regimen
                  Drug  --  Choice                Drug  --  Choice
"most likely"
   ITEM-3         GENTAMICIN -- 3rd       CHLORAMPHENICOL-AND-GENTAMICIN -- 1st
   ITEM-2  AMPICILLIN-AND-GENTAMICIN -- 1st       CHLORAMPHENICOL -- 1st
   ITEM-1         GENTAMICIN -- 2nd               GENTAMICIN -- 2nd

"less likely"
   ITEM-4         GENTAMICIN -- 2nd               GENTAMICIN -- 2nd


(The desirability of a drug is defined to be its lowest ranking for the items
it covers.)
```

Figure 11. Evaluating a User's Choice of Therapy

Both prescriptions include fewer than 3 drugs, so we must look at how highly ranked each prescription is for the most likely organism(s).

Your prescription of 1 first choice drug (AMPICILLIN for ITEM-2) and 1 third choice drug (GENTAMICIN for ITEM-3) is not as good as MYCIN's prescription of 1 first choice drug (CHLORAMPHENICOL for ITEM-2 and ITEM-3) and 1 second choice drug (GENTAMICIN for ITEM-1).

[You may refer to your regimen as RECOMMENDATION-2 in later questions.]

Figure 11. **Evaluating a User's Choice of Therapy** (continued)

Each drug the user suggests for an item is first formed into a standard internal question for CHRONICLER: "Why wasn't <drug> prescribed for <item>?" If there is a negative criterion about this drug for this item in the event history, it is printed and the user is given the option of selecting another drug.

Once the user has supplied a set of drugs to cover for all of the most likely organisms, his proposal is tested for the criteria of drug class uniqueness and patient-specific factors (described in Section 4.4). If the proposal is approved, this recommendation is compared to the program's choice of therapy, just as the program compares its alternatives to its own first choice recommendation [6]. It is also possible to directly invoke the therapy comparison routine [7].

---

[6] The explanations at this point are more pedagogical than those supplied when the program compares its own alternatives. It seems desirable to phrase comparisons as positively as possible to avoid irritating the user.

[7] We plan to use it as part of a computer-aided instruction system based on MYCIN [3].

8 Some Unsolved Problems

There are a number of improvements that might be made to this system. Among the most important to potential users would be a more flexible question format. In our experience, physicians tend to address short, unspecific questions to the program, e.g., "Why Ampicillin?" or "What happened to E.coli?" Processing these questions will require a fairly sophisticated preprocessor that can help the user define his question more precisely, or at least make some plausible assumptions.

Second, we anticipate the need to explain the heuristics which now are only describable in a template form [8]. A user might like to know what a "drug sensitivity" is, or why a heuristic was not used. Providing simple, fixed-text definitions is easy, but discussing a particular heuristic to the extent of explaining why it was not applicable is well beyond the capabilities of this explanation system.

One possible solution is to internally represent the heuristics in a rule-like form with a set of preconditions in program-readable predicates, like MYCIN's rules. We could then say, for example, that a drug was lowered in rank because its sensitivity was "intermediate", though it was a current therapy (which would otherwise be reason for continuing to prescribe it). Thus, we would be splitting a medical criterion into its logical components. Moreover, human explanations sometimes include hypothetical relations that have important instructional benefit, e.g. "if all of the drugs had been "intermediate," then this current therapy would have been given preference." In general, paraphrasing explanations, explaining why an event failed to take

---

[8] That is, each medical heuristic has a string with blanks associated with it, e.g., <drug> "was discounted for" <item> "because it was not DEFINITE that the item was sensitive to this drug."

place, and relating decisions are difficult because they require some representation of what the heuristics mean. But providing a handle on these underlying concepts is a far cry from a system that can only fill in templates.

Third, it is important to justify the medical heuristics and a priori preference ranks for drugs. We now provide text annotations that include references and comments about shortcomings and intent.

Finally, we could further develop the tutorial aspects of the explanation system. Rather than passively answering questions, the explanation system might endeavor to teach the user about the overall structure and philosophy of the program (upon request!). For example, a user might appreciate the optimality of the results better if he understood the separation of factors into local and global considerations. Besides explaining the results of a particular run, an explanation system might characterize individual decisions in the context of the program's overall design.

9 Conclusions

We have developed a system that prescribes "optimal" therapy and is able to provide simple, useful explanations. The system is based on a number of design ideas that are summarized below:

1) Separate the local and global optimality criteria

2) Apply these criteria in comprehensible steps: a generate-and-test control structure was found to be suitable

3) Justify selected therapies by using canonical descriptions that: a) juggle several global criteria at once, and b) permit direct comparison of alternatives

4) Exploit the simple control structure by using a state transition diagram to order retrieval of traces

In addition, the explanation system has benefited from a few simplifying factors:

1) There are small number of traces (fewer than 50 drugs to keep track of and fewer than 25 strategies that might be applied)

2) There is a single basic question: Why was(n't) a particular drug prescribed for a particular organism?

While this therapy selection algorithm may appear straightforward, it was presented to us by the physicians in the form of an unstructured list of factors to consider. The medical experts did not order these considerations, and were not sure how conflicting constraints should be resolved. The framework we imposed, namely invoking optimality criteria locally and globally within a generate and test control structure and describing output canonically, provided a language that enabled us to codify the physicians' judgments, thereby significantly improving the performance and manageability of the program.

Moreover, this well-structured design enables us to print simple explanations of the program's decisions and compare alternative solutions. We have provided this facility because we want the program to be used intelligently. If a user is confused or disagrees with the optimality criteria, we expect him to feel free to reject the results. We hope that the simple explanation system we have provided will encourage thoughtful use of the therapy selection program.

Acknowledgments

The therapy selector was developed by Victor Yu, M.D., Shari Wraith, B.S. Pharm, and the author. Part of the generator was coded by William van Melle. We benefited greatly from experience with Ted Shortliffe's original algorithm. Jim Bennett and Randy Davis made helpful suggestions for revising this paper. [Research sponsored in part by Bureau of Health Services Research and Evaluation, Grant HS01544 and ARPA Contract DAHC15-73-C-0435. SUMEX-AIM computer facilities supported by Biotechnology Resources, Grant RR-00785.]

References

1. Bobrow, Robert J. and Brown, John S. (1975). SYSTEMATIC UNDERSTANDING: Synthesis, Analysis and Contingent Knowledge in Specialized Understanding Systems. In Representation and Understanding Studies in Cognitive Science, Bobrow, D. & Collins, A. (Eds.) Academic Press.

2. Brown, John S., Burton, Richard R., and Zdybel, Frank (1973). A Model-Driven Question-Answering System for Mixed-Initiative Computer-Assisted Instruction, IEEE Transactions on Systems, Man and Cybernetics Vol. SMC-3, No. 3, pp. 248-257.

3. Clancey, William J. (1979). Tutoring Rules for Guiding a Case Method Dialogue. To appear in the Int. J of Man-Machine Studies, January, 1979.

4. Davis, R., King, J.J. (1977). An Overview of Production Systems. Machine Intelligence 8: Machine Representations of Knowledge (eds E. W. Elcock and D. Michie), John Wylie.

5. Davis, R., Buchanan, B., and Shortliffe, E. (1977). Production Rules as a Representation for a Knowledge-Based Consultation Program. Artificial Intelligence Journal, vol. 8 no. 1.

6. Elstein, Arthur S., Shulman, Lee S., and Sprafka, Sarah A. (1978). Medical Problem Solving: An Analysis of Clincal Reasoning. Harvard University Press, Cambridge, Massachusetts.

7. Scott, A. C., Clancey, W. J., Davis, R., Shortliffe, E. H. (1977). Explanation Capabilities of Production-Based Consultation Systems. Amer. J. of Computational Linguistics, Microfiche 62, also available as HPP-77-1, Heuristic Programming Project, Knowledge-based Consultation Systems, Stanford, California.

8. Scragg, Greg W. (1975). Answering Process Questions. IJCAI4, pp.435-442.

9. Scragg, Greg W. (1975). Answering Questions about Processes. In Explorations in Cognition, Norman, D.A. & Rumelhart, D.E. (Eds.) San Francisco: Freeman.

10. Shortliffe, E.H. (1974). MYCIN: A rule-based computer program for advising physicians regarding antimicrobial therapy selection. Ph.D. dissertation in Medical Information Sciences, Stanford University. Also, Computer-Based Medical Consultations: MYCIN, American Elsevier, New York, 1976.

11. Winograd, Terry (1972). Understanding Natural Language, Ph. D. dissertation, Academic Press.

12. Yu, V.L., Buchanan, B.G., Shortliffe, E.H., Wraith, S.M., Davis, R., Scott, A. Carlisle, & Cohen, S.N. (1978). Evaluating the Performance of a Computer-Based Consultant. To appear in Computer Programs in Biomedicine.