Massachusetts Institute of Technology
Artificial Intelligence Laboratory

Working Paper No. 137 December 1976

PSUDOC - A Simple Diagnostic Program

by

Tomás Lozano-Pérez

Abstract

This paper describes PSUDOC, a very simple LISP program to carry out some medical diagnosis tasks. The program's domain is a subset of clinical medicine characterized by patients presenting with edema and/or hematuria. The program's goal is to go from the presenting symptoms to a hypothesis of the underlying disease state. The program uses a variation of simple tree searching strategies called ETS.

This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for the laboratory's research is provided in part by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contract N00014-75-C-0643.

Introduction

This paper describes PSUDOC, a very simple LISP program to carry out some medical diagnosis tasks. The program's domain is a subset of clinical medicine characterized by patients presenting with edema and/or hematuria. The program's goal is to go from the presenting symptoms to a hypothesis of the underlying disease state. This process is sometimes called "taking the present illness" [4].

The reasons for using the computer as an "aid to clinical diagnosis and management" have been outlined elsewhere [2-8]. The issue of how the computer might best be used in a clinical setting is a complex and important one but one that I believe premature. I will therefore ignore it in this paper. Rather, I will concentrate on what I view as a more basic problem. If the computer is to serve in any capacity in a clinical environment, its performance must compare favorably with human performance on some clinical task. Only then can other arguments be advanced for its use. Such programs exist but, traditionally, they have been confined to small, well defined, domains such as antibiotic therapy for bacteremia [1], digitalis therapy [8] and management of acute renal failure [2]. Recently, attempts have been made to deal with broader areas of medical knowledge [4-7]. None of these programs can be characterized as trivial. This fact makes it difficult to answer the question "Why is medical diagnosis hard?". It is difficult because we cannot specify how simple strategies would fail. While intuition might point out some "obvious flaws", intuition is notoriously fickle and unreliable. What is needed is a "baseline program" that can serve as a "lower bound" on complexity and helps isolate problems for further research. That, then, is the aim of this paper.

Before we discuss the requirements on a good baseline program in the field of medical diagnosis, let us try to put the field into some perspective.

Medical Diagnosis as a Research Domain

There is no doubt that medical diagnosis is a vast and complex field. This kind of complexity is forbidding to the beginner and might suggest, at first, a poor choice for a research domain. Closer examination reveals that it offers many advantages as a research domain for Artificial Intelligence. There are clearly defined success criteria; namely, agreeing with the experts. It is easily partitionable; specialization is routinely practiced in the field. It also has the potential of significant relevance to society.

Of greater importance, at least technically, is that the field shares many properties with other subjects of Artificial Intelligence research. Medical diagnosis is a recognition problem; as are vision and speech understanding. Such problems typically involve three conceptual stages: feature detection, hypothesis generation and verification. Work on medical diagnosis can provide insight into the mechanisms needed for integrating these processes in the presence of uncertainty; while avoiding the excessive computational requirements of low-level visual and auditory processing.

The features in medical diagnosis are the symptoms, signs and test results available to the physician. We assume these features to be available symbolically. This is similar to starting visual

recognition with the output of the first phase of a "traditional" line finder [10] (or a "primal sketch" [11]). In this paper, I do not touch on the problem of data acquisition; this domain still remains largely unexplored. We will view the diagnosis problem as that of consulting for a person, not necessarily a physician, but one well trained in recognition of signs and symptoms.

Features in all recognition problems generally have a direct relation between specificity and cost. The cheapest features, presenting symptoms, are usually small in number and thus highly ambiguous. Since costs in medicine are measured in morbidity and mortality, as well as time and money, we want a diagnosis procedure that is efficient, in terms of the number and cost of the features required. Because of this desire to minimize the "expense" and the ambiguity of readily available features, the main tool of a diagnostician (or recognition program) is coherence. This generally involves generating hypotheses that accounts for the available facts and verifying it by requesting additional, confirming, facts. This mechanism has the additional advantage of suggesting features to look for which might be missed in a non-directed search [9]. This is the traditional argument against the use of non-physicians for some diagnostic tasks. I believe it to be a valid one in medicine as well as in the use of knowledge in other recognition tasks.

Medical diagnosis has certain characteristics that set it apart from other AI domains. Because the decisions of a medical diagnosis program may have momentous consequences to individuals, it must be convincing. Not convincing merely in terms of performance, but also in being able to justify its conclusions.

The requirements on a good baseline program

If a program is to serve as a basis for comparison it must fulfill several conditions:

1) Simplicity
2) Flexibility
3) Effectiveness

If it is to be a baseline it must be simple. It should also be easy to change so as to embody some variation in strategies. The program must also be fairly effective. The failings of a totally inept program would provide no insight into the difficulties of the diagnosis process. I will now consider several frequently discussed diagnostic strategies in the light of these reqirements and our previous dicussion of the characteristics of the domain. My goal will be to find some simple strategy that fulfills the three goals stated above.

There are three strategies that are usually mentioned in connection with the medical diagnosis problem.

1) Bayesian decision analysis,
2) Feature intersection, and
3) Tree traversal strategies.

They all have the appeal of simplicity. How they rate on flexibility and effectiveness will have to

decide between them.

Bayesian decision analysis

Uncertainty is prominent in most recognition domains. This explains the fact that probabilistic methods are always among the first applied in all recognition tasks. In the domain of vision, statistical pattern recognition was the earliest approach and still pervades the field. In medical diagnosis, some of the earliest attempts at automating diagnosis were based on decision theory, notably Bayesian decision analysis. Extensive work on the applications of this technique to medical decision making exists [2,3]. The consensus seems to be that it can be made to work on well-constrained problems, but that more general problems severely violate the assumptions on which the methodology is based.

Even if it could be made to work adequately, decision theory leaves something to be desired as a diagnostic mechanism. It provides very weak explanations for its diagnoses. It seems likely that decision analysis will be more fruitful as a tool for selecting treatments or choosing tests than as a means of automated diagnosis.

Feature Intersection

In its simplest form, feature intersection involves finding the set of diseases that can give rise to the collection of known symptoms. In this form, the strategy is clearly not viable as a diagnostic mechanism. Patients normally present with few symptoms. The set of diseases that can account for those symptoms is usually large. Edema is a good example of this. It can be the manifestation of sundry renal problems but also of congestive heart failure, cirrhosis, local circulatory obstruction or malnutrition. Another example is the symptom pair of gross hematuria and hemoptysis. This is considered the hallmark of Goodpasture's syndrome (a deadly disease) but can also be produced (more commonly) by severe trauma such as a car accident. Another problem is that a given set of symptoms might not be due to a single disease state. Trying to find one disease that can manifest them all is hopeless.

Because complete matches are not likely, the feature intersection method requires as its basic operation, a best partial match algorithm. This is one of the hardest problems in computer science. Also, the method seems to be very susceptible to erroneous information. There is no good place where knowledge of likely errors can be stored. The only place seems to be the matching algorithm, which seems the wrong place to put it.

Tree Traversal Strategies

Tree traversal strategies (T-strategies for short) are probably the simplest of the three strategies mentioned above. A simple form of such a strategy is the following:

> Consider a tree whose internal nodes are tests (also questions) with multiple results. For each answer there is a link in the tree joining that node with another node or a leaf. The leaves of the tree are diagnoses. The diagnosis problem consists in

traversing the tree from the root (a presenting symptom) to a leaf (a diagnosis).

In this form, tree traversal strategies have been used to obtain the patient's medical history and presenting complaints. They are in fact so simple that they are never taken seriously as a diagnostic procedure. I will examine some of the objections to the method in detail.

Objections to T-strategies

The two major objection to T-strategies are:

1) Duplication of knowledge
2) Irrevocable decisions

The first objection is prompted by diseases that present in several distinct fashions. Each of these might be a different path through the tree all leading to nodes with a common structure below it. This causes duplication of the same knowledge at many places in the tree. The simplest way of correcting this problem is to give up the tree-structure restriction and allow a more general network structure. There are two problems with this solution. One is the possibility of loops and the other is the problem of maintaining context, e.g. the different presentations might have different likely complications. This last is a very common problem in medicine because disease states are not always defined by etiology but are often classified by clinical or pathological similarities. For example, although Acute Tubular Necrosis (ATN) is a well defined pathological and clinical state it is not an ultimate etiology. Determining the etiology, whether it is due to a nephrotoxic agent such as carbon tetrachloride or due to ischemia (reduced blood supply) caused by shock is an important part of the diagnosis process. So the information common to all forms of ATN must be interpreted within the context of its cause, since this affects the choice of treatment.

The other main objection is prompted by the fact that a T-strategy forces us to make definite decisions at each node based on the result of the question (or questions) asked at that node. This makes it seem susceptible to failure due to ambiguous, uncertain or mistaken results. This need not be the case. Nodes in the tree need not correspond to unique internal states of the individual. Rather they represent states of knowledge of the diagnostic process. As such they are equally suitable for representing certain or uncertain states of knowledge. The problem of coping with ambiguous or misleading data is handled easily. When a conflicting situation is noticed we can take a "side step" in the tree. This amounts to having differential diagnosis links in our structure. Since we have given up the tree structure in favor of a more general network structure, this poses no further problems.

A more difficult problem is handling the possibility of multiple paths for a given result at a node. This is a common result in medicine when tests provide inconclusive evidence. What is needed is a way of ranking the alternatives according to the evidence. We can approach this problem by explicitly assigning a node in the network to each of the possible rank orders. This is feasible for orderings of few hypotheses. Another alternative is to rank the alternatives (i.e. the paths out of a node for a given answer), pursue the most likely one and return to the others later. This leads to an exploration of the network which breaks with the traditional depth-first order. It is closer to the

branch and bound heuristic searches. The branch and bound parallel becomes stronger if we use the result of the test at the node to estimate the probabilities of success along each link and then follow up the most likely link.

I will refer to the T-strategy that incorporates the modifications suggested in the previous paragraphs as an "Extended T-strategy (ETS)". This might be sounding a bit complex but it really is not. I will describe a very simple implementation of a general class of ETS's.

PSUDOC: An inplementation of ETS'S

Consider each node in the network as a simple program. Based on some set of questions or tests it decides what to do next. The repertoire of actions available to each node are:

1) PURSUE a node. This involves calling the procedure associated with that node, unless the node is currently being pursued; in which case it has no effect.
2) Indicate a node is LIKELY. This corresponds to placing the node in a priority queue. Repeatedly placing a node on the queue increases its priority.
3) Indicate a node is UNLIKELY. This reduces the node's priority.
4) DISCARD a node. This removes the node from further consideration. Attempts to pursue the node will immediately return NIL.
5) ESTABLISH a node. This also removes the node from consideration. A PURSUE operation on that node will thereafter immediately return T.

The top level control structure is simply to pursue the nodes associated with each of the presenting symptoms. After this is done, we pursue the node with the highest priority, which has not been previously pursued.

It should be clear how this implementation corresponds to an ETS. The nodes correspond to the basic program chunks mentioned above. The links in the network exist to each of the nodes used as an argument to one of the first four primitives. Loops are avoided by having PURSUE keep a push down list of nodes currently being considered. If a PURSUE action is executed on a node already on this list then the program merely returns.

Each node, when PURSUED, returns either T or NIL. The former indicates that the program succeded in ESTABLISHING the node. Otherwise it returns NIL. A list of ESTABLISHED nodes is kept to avoid repetition. The fact that the node's program returns to the caller (in the time-honored fashion of simple programs) means that we can exploit the way the particular disease presents. This is the solution to the objection that T-strategies are forced to duplicate knowledge to maintain context. It is possible to argue that the binary nature of the outcome of the PURSUE operation is inadequate. In medicine few things are really established before an autopsy is done. Rather, probabilities are determined. This is only a minor change. Most of the nodes perform a probabilistic scoring before deciding to ESTABLISH the node. Instead of the binary decision, the actual score can be returned. This is an example of the ease with which changes can be incorporated into the structure.

Differences in diagnostic style can be easily incorporated into the programs by changing the patterns of PURSUE's and LIKELY's. Favoring the former produces a kind of depth-first or

confirmation style. The latter tends to distribute the search into a style reminiscent of a review of systems.

The major attraction of this approach is its ability to progressively focus in on a particular disease without jumping to hasty hypotheses. Namely, the presence of edema suggests a few broad classes of diseases rather than many specific ones. By keeping to fairly general classes we can ask questions that will prune the tree early in the diagnosis. This also avoids the problem of continuously triggering new hypotheses as the diagnosis progresses. It also provides a more coherent stream of questions.

On the other hand, the mechanisms do not force any particular style. It is equally well suited to jumping to detailed diagnoses. Sometimes this is the right thing to do. PSUDOC is not so much a theory of diagnosis as a mechanism for implementing a wide range of diagnostic styles. This freedom demands discipline on the part of the programmer, but I believe the ease of use is worth the effort.

A Sample PSUDOC Program

This section presents a path through the diagnosis tree starting at edema and ending at Acute Glomerulonephritis. This is, of course, not the optimal representation of the nodes. It is only the best I could come up with. I make no claims as to the accuracy of the medical knowledge embodied in these programs.

One important thing to notice in the programs is the side branches. By this I mean tests that suggest diseases commonly confused with the disease under consideration. Another useful technique is the "circular" use of the PURSUE primitive. Notice that NEPHRITIC can PURSUE AGN and AGN can PURSUE NEPHRITIC. This avoids duplication of information without fear of infinite loops. If some node PURSUEs AGN, it will automatically get the knowledge in NEPHRITIC.

Each of the findings (e.g. PROTEINURIA) is implemented a function which checks that the patient does in fact have the finding with the required properties given as arguments. If the status of the finding is unknown, then the function proceeds to ask for all the available information concerning the particualr finding. It does this in a multiple choice fashion. This question asking strategy is primitive but adequate for experimental purposes. I will discuss alternate strategies in the section on extensions below.

Figure 1 shows the EDEMA node. The program starts out by checking the possibility that the edema has a renal etiology. I've divided kidney diseases into a few broad classes: the NEPHRITIC syndrome, the NEPHROTIC syndrome, renal failure and urinary tract infections (UTI). These don't exhaust the wide range of renal diseases but they are the only ones I half-way understand. The presence of proteinuria is not conclusive evidence for a renal disease but the absence of protein in the urine makes NEPHRITIC disease unlikely. It also rules out the NEPHROTIC syndrome completely. The NEPHROTIC diseases are defined clinically by massive proteinuria.

The next step in the EDEMA program is to check for the possibility of CIRRHOSIS. This test is conditioned on the patient being old or middle-aged. Then three of the common characteristics of cirrhotics are checked: heavy alcohol consumption, jaundiced skin or distended belly (ASCITES). Any of these characteristics in an older person with edema might suggest cirrhosis. In fact, the program looks into it right away (PURSUE). It might simply have been postponed by indicating it was LIKELY.

Independently of what happens to the CIRRHOSIS investigation, the presence of proteinuria would require investigating the possibility of RENAL disease.

If shortness of breath (DYSPNEA) is present with either episodes of waking up with a feeling of not being able to breathe or with postural complications then congestive heart failure (CHF) is very likely and thus PURSUEd.

The program then checks the characteristics of the edema which might favor either a renal or cardiac etiology. This is fairly unreliable evidence so only likelihoods are determined from this. The last two checks in the program test for alternative explanations to the edema. The function

Figure 1

```
(DD EDEMA
    (AND (PROTEINURIA ABSENT)
         (DISCARD NEPHROTIC)
         (UNLIKELY NEPHRITIC))
    (AND (PATIENT (OR OLD MIDDLE-AGED))
         (OR (ALCOHOL-CONSUMPTION (OR HEAVY ALCOHOLIC))
             (JAUNDICE PRESENT)
             (ASCITES PRESENT))
         (PURSUE CIRRHOSIS))
    (AND (PROTEINURIA PRESENT)
         (PURSUE RENAL))
    (AND (DYSPNEA PRESENT)
         (OR (PAROXYSMAL-NOCTURNAL-DYSPNEA PRESENT)
             (ORTHOPNEA PRESENT))
         (PURSUE CHF))
    (AND (EDEMA (NOT (OR ASYMMETRICAL PAINFUL ERYTHEMATOUS)))
         (COND ((EDEMA (NOT (OR PERI-ORBITAL FACIAL)))
                (COND ((EDEMA WORSE-IN-EVENING) (PURSUE CHF))
                      (T (LIKELY CHF))))
               ((EDEMA (OR PERI-ORBITAL FACIAL))
                (COND ((EDEMA WORSE-IN-MORNING) (PURSUE NEPHRITIC))
                      (T (LIKELY NEPHRITIC))))))
    (AND (EDEMA (OR PAINFUL ASYMMETRICAL ERYTHEMATOUS)
                (NOT GENERALIZED))
         (PURSUE CELLULITIS)
         (REEVALUATE))
    (AND (EDEMA FREQUENT) (PATIENT FEMALE) (PURSUE CYCLIC-EDEMA)))
```

REEVALUATE terminates a program and returns NIL.

Figure 2 shows the top-level RENAL node. The main diagnostic tool at this level is the contents of the urinary sediment. Most of this information is used to suggest hypotheses. The only check that really calls for immediate attention is the presence of excessive amounts of nitrogenous wastes (BUN and CREATININE) in the blood. These are indices of kidney function. If wastes are high and/or rising then the possibility of acute renal insufficiency deserves attention.

Below the RENAL node is the NEPHRITIC syndrome (Fig. 3). Its first action is to check for renal failure. This is done because some node might suggest a nephritic etiology directly and bypass the RENAL node. An alternative is simply to PURSUE RENAL from NEPHRITIC. This will remove the need for the first few checks in the program. The function REEVALUATE merely causes a return from the node with value NIL. It is used in situations where the node under consideration is found to be inadequate. It insures that if either HEMATURIA or NEPHROTIC succeeds the consideration of NEPHRITIC will be aborted. Notice that massive edema suggests the nephrotic syndrome but does not warrant rejecting NEPHRITIC.

The next step checks for the evidence that almost defines the nephritic syndrome: hematuria and proteinuria. It then suggests common complications are likely, e.g. hypertension in adults and congestive heart failure for all.

The next stage tries to pin down the etiology to either acute, chronic or focal (benign) glomerulonephritis; which are the main causes of the nephritic syndrome. The version of the program shown is very stubborn about believing that acute glomerulonephritis always shows up in people with a recent strep infection. This is the most common cause but it should probably not be so dogmatic. Notice that an extra question about the time between the onset of renal symptoms and the infection is introduced into the question stream. I believe this ability to introduce questions not directly specified as properties of certain findings is a powerful tool.

The program checks whether the chronic or focal forms of glomerular disease have been established and if so rules out the acute form. The last part of the program checks for the common complication of sodium retention and possible obscure causes of renal symptoms.

The AGN (acute glomerulonephritis) node (Fig. 4) first makes sure that the NEPHRITIC node has been called (by doing a PURSUE on it) and then checks to see if it can clinch the diagnosis. The rest of the program consists of a scoring function taken from the Present Illness Program [4]. Its purpose is to obtain supporting evidence for the diagnosis and weigh the evidence. The scoring method is to run each branch of the function like a LISP COND. Each branch contributes numerical factors that are added up and compared to the maximum possible result. If the ratio is above some threshold then SCORE returns T. To avoid excessive questions, if the score ever gets below a value where the maximum of all the remaining branches would not raise it above the threshold, then the function returns without asking any more questions.

Figure 2

```
(DD RENAL
    (AND (URINARY-SEDIMENT RED-CELLS-IN RED-CELL-CASTS-IN)
         (LIKELY NEPHRITIC))
    (AND (URINARY-SEDIMENT  WHITE-CELLS-IN WHITE-CELL-CASTS-IN)
         (LIKELY UTI))
    (AND (URINARY-SEDIMENT (OR RENAL-CELLS-IN RENAL-CELL-CASTS-IN))
         (LIKELY ATN))
    (AND (URINARY-SEDIMENT (OR FREE-FAT-GLOBULES-IN
                               OVAL-FAT-BODIES-IN
                               FATTY-CASTS-IN))
         (LIKELY NEPHROTIC))
    (AND (OR (BUN MODERATELY-ELEVATED VERY-HIGH RISING)
             (CREATININE MODERATELY-ELEVATED VERY-HIGH RISING))
         (PURSUE RENAL-FAILURE))
    (AND (PROTEINURIA (OR HEAVY >5GRAMS//24HRS))
         (OR (EDEMA (OR MASSIVE 4+ 3+ GENERALIZED))
             (NOT (PRESENTING HEMATURIA)))
         (LIKELY NEPHROTIC))
    (AND (PROTEINURIA PRESENT) (HEMATURIA PRESENT) (LIKELY NEPHRITIC)))
```

Figure 3

```
(DD
 NEPHRITIC
 (AND (OR (CREATININE (OR MODERATELY-ELEVATED VERY-HIGH RISING))
          (BUN (OR MODERATELY-ELEVATED VERY-HIGH RISING)))
      (PURSUE RENAL-FAILURE))
 (AND (PROTEINURIA ABSENT) (DISCARD NEPHRITIC) (REEVALUATE))
 (AND (PROTEINURIA LIGHT)
      (HEMATURIA GROSS)
      (PURSUE HEMATURIA)
      (REEVALUATE))
 (AND (PROTEINURIA (OR HEAVY >5GRAMS//24HRS))
      (HEMATURIA (NOT GROSS))
      (PURSUE NEPHROTIC)
      (REEVALUATE))
 (AND (EDEMA MASSIVE) (PURSUE NEPHROTIC))
 (AND (PROTEINURIA PRESENT)
      (OR (URINARY-SEDIMENT RED-CELL-CASTS-IN) (PRESENTING HEMATURIA))
      (ESTABLISH NEPHRITIC)
      (LIKELY AGN)
      (AND (PATIENT (NOT (OR INFANT CHILD)))
           (LIKELY ACUTE-HYPERTENSION))
      (LIKELY CHF))
 (AND (OR (HEMATURIA (PAST . PRESENT))
          (PROTEINURIA (PAST . PRESENT))
          (EDEMA (NOT FIRST-TIME)))
      (LIKELY CGN FGN NEPHROTIC)
      (UNLIKELY AGN)
      (COND ((CREATININE (NOT NORMAL)) (PURSUE CGN))
            ((HYPERTENSION (OR PRESENT (PAST . PRESENT)))
             (PURSUE CGN))
            ((EDEMA (AND ABSENT (PAST . ABSENT))) (PURSUE FGN))
            ((AND (PURSUE (RECENT-PAST . UPPER-RESPIRATORY-INFECTION))
                  (PURSUE FGN)))))
 (AND (PURSUE (RECENT-PAST . STREP-INFECTION))
      (ASKUSER* (TIME BETWEEN STREP INFECTION AND ONSET OF RENAL SYMPTOMS?)
                (WEEKS ONE-OR-TWO-DAYS HOURS))
      (COND ((EQ ANSWER 'WEEKS) (PURSUE AGN))
            ((MEMQ ANSWER '(ONE-OR-TWO-DAYS HOURS))
             (PURSUE CGN))))
  (AND (OR (ESTABLISHED? CGN) (ESTABLISHED? FGN)) (DISCARD AGN))
  (AND (OR (EDEMA PRESENT
                  (NOT (OR ASYMMETRICAL PAINFUL ERYTHEMATOUS)))
           (EDEMA GENERALIZED)
```

```
          (URINE-SODIUM LOW))
       (ESTABLISH SODIUM-RETENTION))
(AND (FEVER PRESENT)
     (MURMUR PRESENT)
     (PURSUE SUBACUTE-BACTERIAL-ENDOCARDITIS))
(AND (JOINT-PAIN PRESENT) (LIKELY SLE HENOCH-SCHOELEIN-PURPURA))
(AND (RASH (OR MALAR PHOTOSENSITIVE))
     (JOINT-PAIN PRESENT)
     (DYSPNEA PRESENT)
     (UNLIKELY CGN)
     (PURSUE SLE)
     (REEVALUATE))
(AND (RASH (OR PURPURIC PETECHIAL PRESENT))
     (OR (COMPLEMENT (NOT LOW)) (ABDOMINAL-PAIN PRESENT))
     (PURSUE HENOCH-SCHOENLEIN-PURPURA)))
```

PURSUE a node should probably be the rule rather than the exception.

The use of short differential diagnosis tests at one node that suggest an alternative path have an important drawback. If the alternative has already been found to be unlikely, the additional questions will be superfluous. This can be remedied by using a slightly different format. Instead of:

```
(AND TEST TEST (PURSUE X))
```

do

```
(CONSIDER X IF (AND TEST TEST))
```

In this way we could first test if X is plausible before asking the questions. An alternative to this suggestion is merely to include the pre-requisites in the program itself, so that simply pursuing the node would have the same effect. If the node were unlikely, then PURSUE would return immediately. This suggestion has an important drawback. The evidence needed to suggest an alternative might differ according to context. This, presumably, is the part of the import of the differential in differential diagnosis. Placing a fixed set of pre-requisites in the nodes would not capture this distinction.

The programs described above do not provide explanations. This is not a serious problem; merely showing the path of the process through the tree provides a fairly good explanation of why something is being asked or why it was decided. In addition, if each call to PURSUE or LIKELY were given, as an extra argument, a short explanation of the purpose of the call; then this could also be saved and/or printed out. The explanations would also serve as commentary for the programs to simplify the debugging of the knowledge base.

Relation to Other Work

PSUDOC is intimately related to both DIALOG [5,6] and the Present Illness Program [4]. DIALOG divides the diseases into an explicit tree structure but it places certain constraints on the use of the tree. If a descendant from a node is being considered, all the other descendants from that node must be considered also. DIALOG also uses several heuristics to partition the hypotheses and weigh the evidence. These complications make it harder to predict its behavior from looking at the knowledge.

PSUDOC is an offshoot of the Present Illness Program (PIP). This research started out as an attempt at implementing a simpler version of PIP which allowed complex triggers to hypotheses. I believe that PIP has potential as a theory of diagnosis, PSUDOC is merely an attempt at clarifying some issues and providing performance for little cost.

The major theoretical difference between PSUDOC on the one hand and PIP and DIALOG on the other is that both PIP and DIALOG make a distinction between the medical knowledge and the diagnostic knowledge. These two types of knowledge are embodied in these systems as data

and program respectively. The former is meant to be added to, while the latter is not. In PSUDOC there are only programs and no distinction is possible between diagnostic and medical knowledge.

The program structure in PSUDOC bears some superficial resemblance to productions [1]. Most of the entries in the programs are test-action pairs. I believe the similarity is purely superficial. The control structure of the two methods are too different to make the analogy useful.

Conclusions

It is hard to accurately gauge the significance of PSUDOC. I think the level of performance is quite gratifying considering the negligible program complexity. The current implementation has many drawbacks, the effects of which should be pursued. I think this can eventually lead to a better understanding of what mechanisms are essential to a medical diagnosis program.

Bibliography

[1] Davis, R., Buchanan, B. and Shortliffe, E. "Production Rules as a Representation for a Knowledge-Based Consultation Program". SAIL Memo AIM-266, Oct. 1975.

[2] Gorry, A. and Betaque, N. "Automating Judgemental Decision Making for a Serious Medical Problem". Management Science, Vol. 17, No. 8, April 1971.

[3] Gorry, A. and Barnett, O. "Sequential Diagnosis by Computer". JAMA, Vol. 205, No. 12, Sept 16, 1968.

[4] Pauker, S., Gorry, A., Kassirer, J. and Schwartz, W. "Towards the Simulation of Clinical Cognition". Preprint.

[5] Pople, H., Myers, J. and Miller, R. "The Dialog Model of Diagnostic Logic in Internal Medicine" Fourt Int. Joint Conference on Artificial Intelligence, 1976.

[6] Pople, H. "Artificial Intelligence Approaches to Computer Aided Medical Consultation", IEEE Intercon Conference, 1975.

[7] Rubin, A. "Hypothesis Formation and Evaluation in Medical Diagnosis", MIT Artificial Intelligence Laboratory Technical Report 316, January 1975.

[8] Silverman, H. "A Digitalis Therapy Advisor", MIT Project MAC, Technical Report 143, January 1975.

[9] The Johns Hopkins Textbook of Medicine.

[10] Horn, B. K. P., "The Binford-Horn Line-finder". MIT Artificial Intelligence Lab., Memo 285, December 1973.

[11] Marr, D. "Early Processing of Visual Information". MIT Artficial Intelligence Lab. Memo 340, December 1975.