Knowledge-Based Systems

Peter Szolovits*
MIT Laboratory for Computer Science

May 12, 1995

Embedding knowledge is a popular and effective means of increasing the power of sophisticated computer applications. While the intellectual roots of this method go back to the late 1960's, the ideas were first codified, systematized and simplified in the 1970's, and have led to a large and successful expert systems industry in the 1980's. Despite these successes, most types of expertise are still extremely difficult to capture, and many fundamental scientific and engineering challenges remain to this field.

This paper will briefly review the origins and motivations for the field, indicate the considerable successes it has achieved, outline the many remaining difficulties, and highlight a few individual research results that point the way to its future development.

1 Expertise

What is expertise? To get an appreciation for the answers to this question, we can turn to one of the great experts in English literature, Sherlock Holmes:

> "It is simplicity itself," said he; "my eyes tell me that on the inside of your left shoe, just where the fire-light strikes it, the leather is scored by six almost parallel cuts. Obviously they have been caused by someone who has very carelessly scraped round the edges of the sole in order to remove crusted mud from it. Hence, you see, my double deduction that you had been out in vile weather, and that you had a particularly malignant boot-slitting specimen of the London slavey."
>
> — A. Conan Doyle, *Sherlock Holmes*, A Scandal in Bohemia

What is it that can account for Holmes' outstanding analytical ability? Certainly, he is capable of brilliant, long, perhaps convoluted logical chains of inference. In this simple case, parallel cuts in Watson's shoe may be explained by scraping, which implies that they were muddy, thus Dr. Watson must have been about in bad weather. But probably more important to Holmes' capacity is his wealth of knowledge about the world. He must be able

*This work has been supported (in part) by the National Institutes of Health, grant number 5 R01 LM 04493 from the National Library of Medicine. This paper was published as a book chapter: P. Szolovits. Knowledge-Based Systems. In A. R. Meyer, J. V. Guttag, R. L. Rivest and P. Szolovits, editors, *Research Directions in Computer Science: An MIT Perspective*, pages 317–370, MIT Press, 1991.

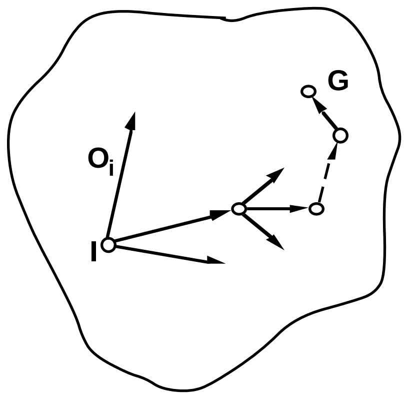


Figure 1: Search space for a robot planning system. I represents the initial state from which search begins, arcs outward from any node indicate possible operations, O_i, that may be done there, leading to the state at the end of the arc, and G is a predicate that tells whether a given state is acceptable as the goal of search.

to make accurate and detailed observations, or else the informative slits in the leather would escape his notice. He must be able to interpret these signs and symptoms appropriately, within a context of immense knowledge about what is likely and what implausible. He must understand customary actions and behavior, habits and social relations. In short, his mind is alive with a vivid and crowded model of the world he seeks to understand.

It is the ability to represent and use such vast and complex interrelationships of real-world knowledge that is the goal of knowledge-based systems, and that sets them apart from other computer programs. In this paper, we begin by making an argument, first recognized in the 1960's, that encoding a great deal of this real-world knowledge is essential to enable computer programs to behave intelligently. We then describe a few simple architectures for programs that can exploit large bodies of knowledge encoded in stylized forms, and point out the success that such programs have had in the commercial marketplace. Next, we turn to a number of nagging problems that indicate a need for deeper analysis, and illustrate some of the promising directions of work by showing some current experimental systems that use novel methods to encode or use knowledge. Finally, we comment on the practice of "knowledge engineering" and the further commercial prospects of expert systems.

2 The Case for Knowledge

How would a robot respond in the morning to hearing its alarm clock ring?[1] Conventional artificial intelligence (AI) approaches to such a problem would suggest that the robot should first make a plan: for example, first roll out of bed, then walk to the clock, and hit the "off" button. Such plans are ordinarily made by a process of search through a vast space of possible alternative plans. Figure 1 illustrates a typical view of a search-based problem solving process. We start in the initial situation, I, and have available to us a set of possible

[1]This example was first suggested to me in a conversation with Joel Moses in the late 1970's.

operations, O_i, each of which leads to a new possible situation, from which we have various further possible operations available. In addition, there is a goal predicate G that tells us whether a given situation in fact satisfies our goal. In the alarm clock example, the initial situation is the robot lying in bed, with the alarm clock ringing, and various other possibly-relevant facts about the scene (e.g., the locations of the bed, clock, intervening obstacles, whether the bed is on the floor or a bunk, ...). Operators include those needed to implement the above plan: getting out of bed, walking, pushing buttons on the alarm clock. Typically, they also include others with possible relevance to the current task, such as throwing light objects, avoiding disturbing noises by covering the ears, etc., as well as those with no obvious relevance, such as making a phone call, eating a meal, etc. Our goal would be that the alarm clock no longer be disturbing the robot.

A particularly simple planner might, for example, begin with our initial scene and consider, in turn, those situations that would result from taking each possible action (operation) from that scene. Because none of them leads directly to the achievement of the goal, it could then consider further situations resulting from new operators applied to those situations, and so on until a goal state was reached. Clearly, if at each step there are k available operations, there will be k^n n-step plans to consider, which is quite impractical for even modest values of k and (especially) n.

Unfortunately, this simple planning-oriented view of problem-solving can degrade even further as we consider more details in a problem. For example, our hypothetical plan begins with the intent to roll out of bed. But in fact, how are we to accomplish this? We may have to consider numerous means of getting off the bed. And having selected a possible method, don't we in fact have to worry about whether we are capable of implementing that method in detail? For example, are our robot's muscles strong enough? Is it possible that our robot's side will be crushed by its weight as it begins to roll? Thus when we solve problems by explicit planning, we are in the counterintuitive situation where apparently adding more knowledge about the domain makes reasoning harder.

Human thinkers fail to be paralyzed by such possibilities by having built up a large repertoire of physical and mental actions that they apply seemingly without thinking—unless some feature of the problem tips them off to a special need for care. Note that this use of *heuristics* is far from guaranteed. Indeed, the floor may have given way and I might fall through by blithely striding toward the clock, my muscles may in fact have atrophied during the night from a yet-undiscovered rapidly-progressive degenerative disease, or a vagrant satellite may be about to rip a swath through my house and destroy the alarm clock, leaving no need for my intervention. Thus, any aspect of the problem is subject to changes that are sufficiently drastic that only a careful re-evaluation of my situation, capabilities and goals will lead to reasonable behavior. Nevertheless, in the vast majority of cases, we act without such re-evaluation.

Fortunately, though virtually nothing of what we "know" is certain, much of it is fairly reliable. Furthermore, as argued above, we cannot afford the alternative—to figure out everything from first principles. Indeed, an intelligent agent that always questions its own mental operations is likely to be in deep trouble, and we identify such a disturbance as psychosis. Thus, a wealth of routine knowledge and the faith to apply it (mostly) uncritically is necessary for everyday life.

Large stocks of knowledge are essential in technical fields as well as in routine life. This

```
          2          3          3                     2
(D36)  (3 X  + 2) SIN(X) + 3 (X  + 2 X + 1) COS(X) SIN (X)

(C37) INTEGRATE(%,X);

             3                    2
          (9 X  - 6 X) SIN(3 X) + (9 X  - 2) COS(3 X)
(D37) 3 (- --------------------------------------------
                             108

              3                    2
   (162 X - 27 X ) SIN(X) + (162 - 81 X ) COS(X)
 + -------------------------------------------
                        108

                                                     3
   3 X SIN(3 X) + COS(3 X) - 9 X SIN(X) - 9 COS(X)   SIN (X)
 - --------------------------------------------- + -------)
                        18                           3

                         2                               2
   6 X SIN(3 X) + (2 - 9 X ) COS(3 X) - 162 X SIN(X) + (81 X  - 162) COS(X)
 - -----------------------------------------------------------------------
                                    36
        3
     COS (X)
+ 2 (------- - COS(X))
        3

. . .

(C40) TRIGSIMP(%);

         3               3
(D40)  (X  + 2 X + 1) SIN (X)
```

Figure 2: Example of an interaction with MACSYMA. Command `C37` says to integrate the formula `D36`, and command `C40` says (after two steps not shown here) to use trigonometric identities to simplify the result, yielding `D40` as the integral of `D36`.

is because creativity is actually relatively rare, and difficult. Therefore, knowing how to do something is far better than being able to figure it out. Most of what we know in science, engineering, business, medicine, law, architecture, etc., does not derive from personal invention but from being taught and shown.

3 Early Expert Systems

Two research groups, both facing difficult real-world technical problems, independently recognized the need to incorporate large amounts of knowledge into programs in the mid- to late 1960's. The Mathlab group at Project Mac, whose principal leaders at that time were Bill Martin, Joel Moses and Carl Engelman, began the development of a powerful, comprehensive system for symbolic mathematical manipulation, which became MACSYMA. Unlike most of its predecessors, which tended to focus on one part or another of the symbolic manipulation task, MACSYMA provides a broad range of capabilities, including integration, power series, various forms of simplification, and support for manipulating matrices, tensors, etc. Figure 2 shows a simple example of an interaction with the system. As is the case with much of the best "expert systems" work to follow, these researchers were less concerned with fitting their efforts into a neat AI paradigm than with taking a credible cut at their problem. Moses, in his doctoral thesis describing the symbolic integration program SIN, states their manifesto:

> "We ... intended no ... study of specific problem-solving mechanisms, but mainly desired a powerful integration program which behaved closely to our conception of expert human integrators."
>
> "Our emphasis in SIN is on the analysis of the problem domain. ... When SIN is solving ... difficult problems, [most notable is] how quickly SIN usually manages to decide which plan to follow and the straightforward manner with which it obtains the solution thereafter."

SIN followed a three-stage strategy, derived from a careful analysis of the problem of symbolic integration. First was a powerful single method, an elaboration of integration by parts, that solved most common problems directly. Second, and most importantly, SIN tried a set of eleven highly-specific methods that attempted to recognize troublesome features of the problem that prevented the first method from working, and then tried to fix those features locally. For example, failure might have been caused by a complex term under a radical; in this case, SIN would try applying a specific method to transform that term into a form amenable to further processing. Third, and finally, SIN would resort to a general-purpose problem solver—one that searched a set of possible solution paths. This last stage only rarely came into play, and when it did, only rarely helped; if the special "tricks" failed, it was unusual for the general-purpose methods to succeed. Thus, perhaps half of SIN's power came from analyzing the problem of integration and determining that the first method was very frequently adequate, and most of the other half came from the second stage tricks, which reworked problems that had (perhaps temporarily) escaped solution by the first.

SIN became part of MACSYMA [8] and continues to play an important role in that system. Later progress in symbolic integration led to the development of the Risch algorithm, which integrates any integrable function in a broad class, and much of this capability is now included in MACSYMA. Nevertheless, the SIN-like first and second stage strategies remain because if they work, they generate more compact solutions. The system as a whole has grown more and more comprehensive, as a user community of thousands of researchers has developed and itself contributed to its significant, sometimes idiosyncratic, but highly useful expansion. The 1986 Macsyma Reference Manual, for example, defines the function

> `bdvac()`: generates the covariant components of the vacuum field equations of the Brans-Dicke gravitational theory ...

which is clearly of use to researchers in a rather narrow domain.

In a retrospective analysis of MACSYMA's success, Bill Martin suggested [17] that building a knowledge-based system consists of identifying and integrating four categories of contributions:

1. *Powerful ideas:* Any system will have, at its core, a small number (perhaps less than ten) of outstanding ideas. These are usually rather general, such as the notion of *recursion*, or, indeed, the *analyze/treat special cases* architecture of SIN.

2. *Great Tricks or Facts of Nature:* What makes a system "expert" is the ability to exploit some techniques of power within its domain. There may be a few tens of such "tricks." The Risch algorithm and Euclid's algorithm would certainly qualify.

3. *Unavoidable Engineering Decisions:* Typically, there will be one or two hundred significant engineering decisions that must all be made with a reasonable degree of harmony in order to make the system elegant, uniform and usable. An example might be the choice in a symbolic manipulation system whether to treat unary minus as a special case or whether always to represent expressions of the form $-x$ as if they were $0 - x$. Such decisions may not, in themselves, seem very critical, but they have far-ranging consequences for many other aspects of the system. The unary/binary choice for negation, for instance, can have a big impact on the design of the simplifier.

4. *Avoidable Engineering Decisions:* Martin might well have called these *postponable* rather than avoidable, but they are the myriad detailed decisions made in the course of fleshing out the capabilities of a large system. MACSYMA at around its tenth anniversary contained over three thousand individual LISP procedures, each embodying several detailed design decisions.

Of course it is important to pour in the knowledge roughly in the order of categories given above. The powerful ideas and great tricks define the overall approach to a problem domain and what a program can hope to accomplish, and engineering decisions—whether unavoidable or postponable—make sense only within those confines. Unfortunately, this lesson has often been lost in later systems that emphasize the value of uniformity over careful analysis and organization; this is a topic we take up again later.

The other major early expert system came from the collaborative efforts of a Stanford University group headed by Joshua Lederberg and Ed Feigenbaum. DENDRAL's task was to determine the three-dimensional structure (isomer) of a chemical compound from its chemical formula, a mass spectrum, and perhaps other information such as nuclear magnetic resonance (NMR) data. Figure 3 shows a schematic mass spectrometer. At the left, the unknown is fragmented, and the fragments are ionized and injected into a magnetic field. The more massive any fragment, the less its trajectory is curved, therefore the further it travels before registering in an array of detectors at the bottom. The number of fragments detected at each mass yields the mass spectrum of the compound.[2]

The naive approach to this task might be to catalog all known mass-spectra and then attempt to match each observed spectrum to all the known ones. Unfortunately, the required degree of accurate and selective matching is probably impossible to achieve, and even if it were, this method would fail for any compound not in the program's library. Furthermore, the relationship between structures and mass spectra is not arbitrary, but is largely predictable from an analysis of how chemical compounds fragment in a mass spectrometer. Merely listing empirical associations between structures and spectra fails to exploit this source of regularity in the domain, and thus makes such a naive program much weaker than necessary.

A significant improvement to this scheme formed the initial starting-point for the DENDRAL team. They observed that if one knows the empirical formula for an unknown, one can

[2]This is a rather naive description of mass spectrometry. For more details on this technique and on DENDRAL itself, see the retrospective volume on that project [15].

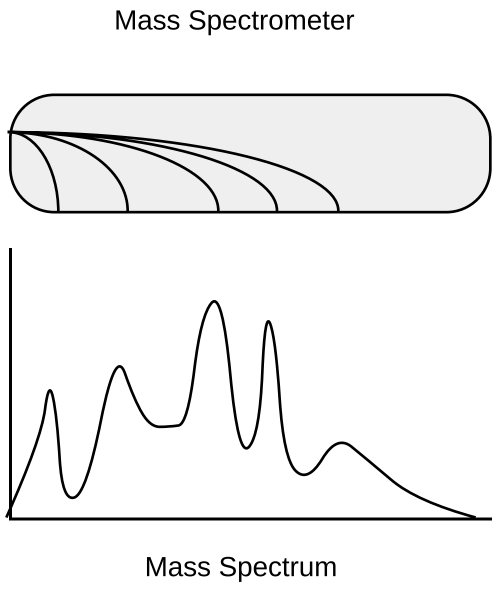


Figure 3: A Mass Spectrometer and the Mass Spectrum input to DENDRAL. The mass spectrometer fragments a chemical compound into constituent parts, accellerates them, and deflects them in inverse proportion to their mass. The result is a mass spectrum, a graph showing for each mass value how many fragments with that mass were detected.

enumerate all the possible three-dimensional structures into which that collection of atoms could possibly be organized.[3] Next, by modeling the fragmentation process in the mass spectrometer, it is possible to predict which bonds in any such structure are likely to break, and thus to predict the sizes of the fragments that should be seen in the spectrum. Therefore the problem of identifying the structure of an unknown with a given formula reduces "merely" to generating all possible structures for that formula, simulating the effects of the mass spectrometer on each such structure, and comparing the predicted mass spectrum to the one observed for the unknown. It is as if the program were generating a custom library for each unknown. Note that this is far more flexible than the original scheme. Unfortunately, it is also computationally quite intractable, because the number of distinct structures possible for an unknown compound may run into the millions.

Nevertheless, human experts working at this task were able to identify unknown compounds, clearly without examining a myriad possible structures. Studying their methods, it

[3]Incidentally, one of the major contributions to chemistry from this project was the development of this generator, called CONGEN, or *constrained generator*. Its main advances were that it could generate all of the often vast number of isomers without repeating any, and that it could take constraints into account. Thus, if the user was willing to posit a given structure for some portion(s) of the molecule, CONGEN would only generate those complete structures that included the posited portions. Significantly, this was done intelligently: it did not simply generate all possibilities and cast out those that did not include the known portions. Instead, it actually used those known portions as the starting point for generating the possible total structures. This made it quite efficient. Not only did this method represent an advance in theoretical chemistry, but it also provided a computational tool of great value quite independent of the DENDRAL system.

Look for two peaks at x_1 and x_2 such that

1. $x_1 + x_2 = M + 28$
2. $x_1 - 28$ ($= x_1'$) is a high peak
3. $x_2 - 28$ ($= x_2'$) is a high peak
4. at least one of x_1 or x_2 is high.

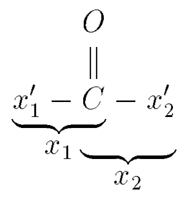


Figure 4: A DENDRAL *specialist* for recognizing a ketone. The conditions correspond to fragmentation of the chain on either side of the $C = O$ structure.

Compound	*No. of isomers*	*Mass spec.*	*NMR*
Di-iso-pentyl ether	989	18	7
Di-n-hexyl ether	6,045	125	2
Bis-2-ethylhexyl ether	151,375	780	21
Di-n-decyl ether	11,428,365	22,366	1

Table 1: Number of isomers generated using no data, mass spectrometer data only, and mass spectra plus NMR data, (excerpted from [5]).

seemed that rather than beginning by enumerating possible structures, the human expert started by looking in the observed spectrum for evidence that certain substructures were present in the unknown. The expert would then consider only those overall structures that included these observed substructures. Similarly, the program was designed [5] to

> "use whatever specialized knowledge and processes and whatever auxiliary data are available to infer pieces ... of the solution. ... For the remaining atoms, ... use the general structure-generating machinery."

The program thus contained a large number of *specialists*—small program fragments that examined the mass spectrum for relevant clues and proposed various substructures. These then contained much of the expert knowledge of the system and led to performance dramatically faster than that of the naive approach. An example of a specialist for detecting a ketone is shown in Figure 4. Because the chain is likely to break at either side of the carbon double-bonded to the oxygen, we expect to see peaks corresponding to the CO being attached to either half of the broken chain, and other peaks corresponding to those halves without the CO. Though presumably such evidence could be mimicked by an alternative, more complicated explanation, expert chemists, like Sherlock Holmes, are willing to take the chance of being misled by such outstanding and valuable clues. So is DENDRAL.

The specialists and the architecture that exploits their observations make an enormous difference. Table 1 (from [5]) summarizes a few of the delightful results of this technique, showing the number of total possible isomers, the number consistent with constraints suggested only by mass spectrum specialists, and the number consistent with constraints from both mass spectrum and NMR specialists.

In the most dramatic demonstration of the value of this technique, note the eleven-million-fold reduction of search for the case of Di-n-decyl ether. Even when search cannot be eliminated, its reduction by orders of magnitude makes possible a simulate-and-match solution to the reduced problem.

4 Expert Systems Architectures

The success of these early expert systems suggested that one should be able to build other such systems for other domains based on the same fundamental ideas. Neither MACSYMA nor DENDRAL were built in a way that was easy to generalize, however, and it quickly became obvious that if expert system construction were to be made a more routine activity, we would have to invent standard ways of capturing and combining knowledge. The goal was to provide the expert system builder with a fixed architecture and to require only that the appropriate knowledge be added. Over the course of a half-dozen years of experimentation with particular systems, mostly in the medical decision-making domain, researchers identified a few apparently useful such general architectures:

- Working backward from a goal to its prerequisites by chaining of simple production rules.
- Matching a pattern of observables to predictions of alternative (clusters of) models.
- Responding to the time-course of a changing system.

4.1 Rule-based Systems

Probably the most influential of these early systems was MYCIN, whose task was the diagnosis and prescription of drugs for bacterial infections of the blood (bacteremia), and later for meningitis as well. Mycin's organization rested on the use of a collection of modular if-then rules, the uniform use of backward chaining to trace back from goals to supporting arguments, and a model of probabilistic inference called *certainty factors* [24] inspired by Carnap and later shown to be a variant of Bayesian inference with a particularly strong independence assumption [12]. Figure 5 shows a rule typical of MYCIN's knowledge base. This rule is particularly simple in that it requires only knowledge of various characteristics of the organism under consideration to derive another of its characteristics, its identity.

Figure 6 requires a somewhat more complex method of interpretation because it refers to not just a single entity but also to a culture. In general, there may be many cultures and many organisms being considered, but it only makes sense to apply a rule to those cultures and organisms where the organism was grown out of that particular culture. MYCIN therefore introduces a *context mechanism*, depicted in Figure 7, that helps control the matching of rules only with appropriate bindings of related variables. It assures that another rule, which might mention a culture, an organism and a drug being used to treat it, can be applied to only the three sets of related entities: {`culture-2, organism-2, drug-1`}, {`culture-2, organism-2, drug-2`} and {`culture-3, organism-4, drug-3`}.

The beginning of a typical MYCIN consultation is shown in Figure 8. After collecting initial data about the patient, including name, age and sex, the program sets as its goal to

```
IF the organism
    1) stains grampos
    2) has coccus shape
    3) grows in chains

THEN
    There is suggestive evidence (.7) that the identity of the
    organism is streptococcus.
```

Figure 5: A MYCIN rule relating characteristics of an organism to its identity.

```
IF
    1) The site of the culture is throat, and
    2) The identity of the organism is streptococcus

THEN
    There is strongly suggestive evidence (.8) that the subtype
    of the organism is not group-D.
```

Figure 6: A MYCIN rule requiring contexts for its proper interpretation. Context must assure that the organism and the culture mentioned in the rule are in fact related; i.e., that the organism grew out of that culture.

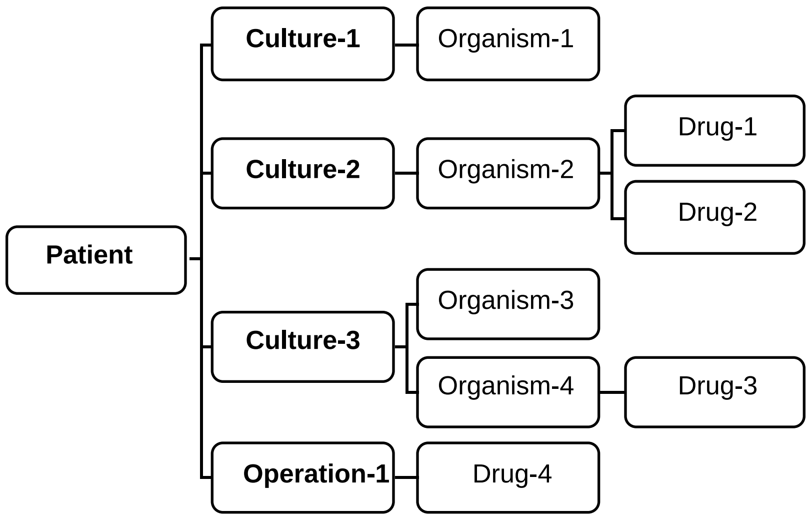


Figure 7: A typical MYCIN context, showing the relationships between the patient, cultures, organisms and drugs.

```
--------PATIENT-1--------
1) Patient's name:  FRED SMITH
2) Sex: MALE
3) Age: 55
4) Have you been able to obtain positive cultures from a site at
   which Fred Smith has an infection? YES
   --------INFECTION-1--------
   5) What is the infection? PRIMARY-BACTEREMIA
   6) Please give the date when signs of INFECTION-1 appeared. 5/5/75
   The most recent positive culture associated with the primary-bacteremia
   will be referred to as:
   --------CULTURE-1--------
   7) From what site was the specimen for CULTURE-1 taken? BLOOD
   8) Please give the date when this culture was obtained. 5/9/75
   The first significant organism from this blood culture will be called:
      --------ORGANISM-1--------
      9) Enter the identity of ORGANISM-1. UNKNOWN
      10) Is ORGANISM-1 a rod or coccus (etc.)? ROD
      11) The gram stain of ORGANISM-1: GRAMNEG
      . . .
```

Figure 8: Beginning of a MYCIN consultation. After determining the patient's name, sex and age, MYCIN begins to identify the relevant objects for its investigation, and to ask questions by chaining backward via its rules from the fundamental goal: determine if there are any significant organisms requiring treatment present in this patient.

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

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

Figure 9: MYCIN's therapy recommendation. Recommendations with the fewest number of drugs that nevertheless cover for all the suspected organism identities are preferred, but other, lower-ranking recommendations may also be displayed.

determine whether there are any significant infections present in this patient that require treatment. Then it hands control over to its general backward-chaining mechanism. In this case, that will seek rules in the knowledge base that can conclude whether there is a significant infection present. Among the premises of such a rule might be attributes of the culture from which an organism was drawn, as well as attributes of the organism, the patient, any currently-active therapies the patient is receiving, etc. In seeking any of these additional data, the program can initiate further backward chaining, tracing through the rules capable of confirming or denying each of these subsidiary questions in turn.

In general, backward chaining follows a highly recursive pattern:

- To find out a fact:
  - If there are rules that can conclude it, try them.
  - After the rules (if any) have been tried and the fact is still not known, ask the user.
- To "try" a rule:
  - Try to find out if the facts in the premises are true.
  - If they all are, then assert the conclusion(s), with a suitable certainty.

In addition, the context mechanism will cause MYCIN to ask about the presence of additional cultures, organisms or drugs if premises of a rule refer to such an object that does not yet exist in the context.

After reaching a diagnostic conclusion, MYCIN turns to making a therapeutic prescription (Figure 9) based on trying to minimize the number of antimicrobial agents needed to do a good job of knocking out the most likely and most dangerous suspected infecting organisms.

Operation of this general scheme with MYCIN's stock of four to eight hundred rules[4] led to behavior that was arguably indistinguishable from that of expert physicians and that was often clearly superior to that of lesser-trained physicians [33]. In addition, by keeping track of what rules were run in which situations, the program can generate moderately sophisticated explanations of its reasoning. Figure 10, for example, shows a trace of how one particular rule, `RULE163`, was used but turned out not to be relevant, and why the system did not reach a conclusion that the questioner might have considered plausible. Our intuition has always been strong that such abilities of programs to explain and justify their behavior would turn out to be critical to their acceptance [1, 27], and more recent surveys of physicians' attitudes support this intuition [30].

4.2 Frame Matching

For diagnostic reasoning, the simplest heuristic is: "it is what it looks like." Figure 11 suggests what happens when a description of the facts about a case is overlaid onto a description of an hypothesis. If the hypothesis is appropriate, there will be a reasonably good fit, though typically some predictions made by the hypothesis will not be found in the patient data, and

[4]The knowledge base grew with time, as the domain was extended to include meningitis.

```
** Did you use RULE 163 to find out anything about ORGANISM-1?

RULE163 was tried in the context of ORGANISM-1, but it failed because it is not
true that the patient has had a genito-urinary tract manipulative procedure
(clause 3).

** Why didn't you consider streptococcus as a possibility?

The following rule could have been used to determine that the identity of
  ORGANISM-1 was streptococcus: RULE033

But clause 2 (the morphology of the organism is coccus) was already known to
  be false for ORGANISM-1, so the rule was never tried.
```

Figure 10: Explanation from a MYCIN consultation. The first question investigates how a particular rule was (or, in this case was not) applicable to a case. The second explores why MYCIN did not reach a particular conclusion.

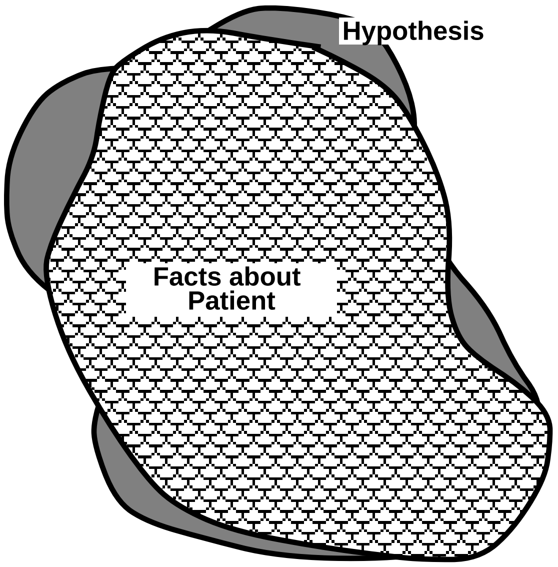


Figure 11: Pattern matching as a basis for diagnostic reasoning. In the absence of a perfect overlap between a case and an hypothesis, differences may be used to drive information acquisition.

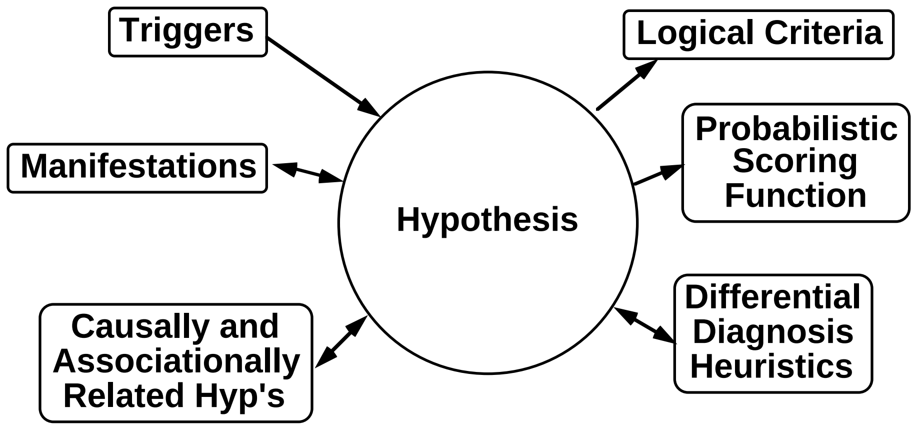

Triggers
Logical Criteria
Manifestations
Probabilistic Scoring Function
Hypothesis
Differential Diagnosis Heuristics
Causally and Associationally Related Hyp's

Figure 12: PIP's components and information flows. The knowledge base is organized around hypotheses. Triggers and other manifestations may evoke hypotheses, which may, in turn, suggest asking about other manifestations. Logical criteria and a scoring function help determine whether an hypothesis should be accepted or rejected, and how strongly it should be pursued. Links to other hypotheses include a set of causal relations and also specific heuristics for how to change the program's focus of attention when it believes it has been misled by earlier evidence.

those data may contain features not explained by the hypothesis. These discrepancies can be used as a good source of guidance to help collect needed additional information to reach the correct diagnostic conclusions efficiently.

The Present Illness Program (PIP), built by our group in the early 1970's [22], embodied this view:

1. From the patient's initial complaints, *guess* a suitable hypothesis.

2. Use the current active hypotheses to guide questioning.

3. Failure to satisfy expectations is the strongest clue to a better hypothesis. Concentrate on the traditional notion of *differential diagnosis*, a means of rapidly shifting attention from an incorrect hypothesis to a better, related one when discrepant information arises.

4. Hypotheses are *activated, de-activated, confirmed* or *rejected* based on (a) logical criteria and (b) probabilities resulting from the presence or absence of findings local to an hypothesis and causal relations to other hypotheses.

Figure 12 shows the information flows centered around hypotheses in PIP. *Triggers*, or strongly evocative findings, implement the guessing strategy. Thus, an important and specific fact that arises early in the consultation can immediately focus the program's attention to the right problem. Once an hypothesis is active, it can suggest others of its expected manifestations as reasonable foci for the program to question the user about. Further confirmation for an hypothesis can also come from the program's co-ordinated beliefs in the

```
NEPHROTIC SYNDROME, a clinical state
FINDINGS:
  *1. Low serum albumin concentration
   2. Heavy proteinuria
  *3. >5 gm/day proteinuria
  *4. Massive symmetrical edema that is painless and not red
  *5. Facial or peri-orbital symmetric edema
   6. High serum cholesterol
   7. Urine lipids present
IS-SUFFICIENT:  Massive pedal edema & >5 gm/day proteinuria
MUST-NOT-HAVE: Proteinuria absent
SCORING . . .
MAY-BE-CAUSED-BY:  AGN, CGN, nephrotoxic drugs, insect bite,
   idiopathic nephrotic syndrome, lupus, diabetes mellitus
MAY-BE-COMPLICATED-BY:  hypovolemia, cellulitis
MAY-BE-CAUSE-OF:
   sodium retention
DIFFERENTIAL DIAGNOSIS:
   neck veins elevated -> constrictive pericarditis
   ascites present -> cirrhosis
   gross hematuria -> renal vein thrombosis
```

Figure 13: PIP's frame for nephrotic syndrome. Among the findings, an asterisk marks triggers. The scoring clause has been suppressed to save space.

presence of related hypotheses. PIP made its decisions to accept or reject an hypothesis either on the basis of logical criteria—the most effective method, when available—or by a complicated probabilistic scoring function that rewarded for the presence of expected findings and expected related hypotheses and penalized for their absence.

Because PIP's underlying focus of attention was strongly influenced by the guesses it made based on triggering from features of the input,[5] it was important for the program to have a graceful way of recovering from an incorrect guess. Our studies of expert human problem-solving in medicine [13] suggested that simple backtracking is not what expert physicians do in this case. Instead, they appear to have "compiled" a set of recovery routines that say "Ah, if I've been misled into pursuing hypothesis x and the following discrepancies arise, this means I should be pursuing y instead." It may well be that learning these recovery heuristics is an important key to the difference between the expert and novice decision-maker. Having them at hand allows the expert to make more rapid guesses, possibly solving a problem very efficiently, because he or she knows that if the guess turns out to be problematic, there are probably good ways to recover from it. The novice, by contrast, must explore hypotheses more thoroughly and systematically, for fear of getting stuck at a bad hypothesis and having nowhere to turn except to start over from the beginning.

Figure 13 shows PIP's model of nephrotic syndrome. Note the differential diagnosis clues at the bottom. For example, if this hypothesis is being considered and evidence arises

[5]This is somewhat reminiscent of DENDRAL.

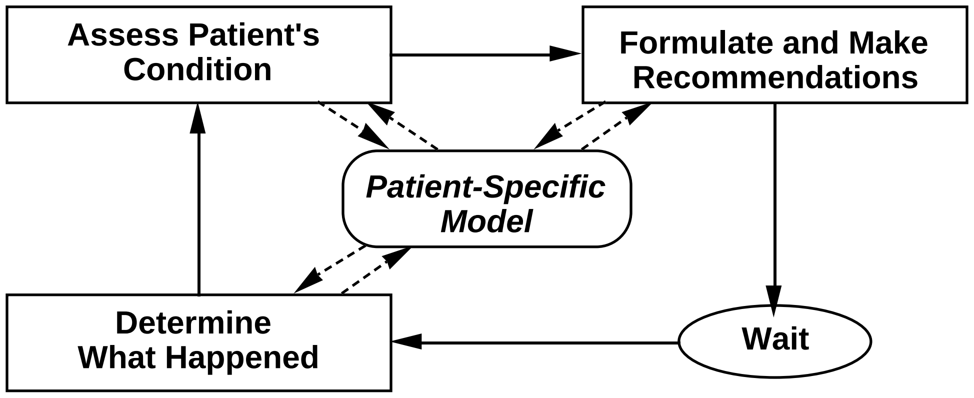

Assess Patient's Condition
Formulate and Make Recommendations
Patient-Specific Model
Determine What Happened
Wait

Figure 14: Feedback loop in therapy in the Digitalis Therapy Advisor. Solid arrows represent the flow of control, cycling clockwise around the therapy loop. Dashed arrows represent data flow to and from the patient-specific model.

for ascites (fluid accumulation in the abdomen), then cirrhosis should be considered as an alternative.

PIP was the first expert system to break into the medical literature [22], and demonstrated reasonable performance and interactive style in its narrow domain of specialty. The INTERNIST-I program [19] developed at the University of Pittsburgh at around the same time used similar techniques, though with more of an emphasis on the systematic sorting of findings and their relations to competing and complementary hypotheses rather than the guess/correct scheme of PIP. The knowledge of INTERNIST-I, which has been developed extensively through the massive efforts of Dr. Jack Myers and his associates, is now also used in a medical educational program called QMR (Quick Medical Reference) [18]. Unlike PIP and INTERNIST-I, both of which guide the interaction to help elicit data that seems most useful to confirm or deny hypotheses under consideration, QMR takes a nondirective approach. It helps to interpret the significance of data reported to it, to compare various hypotheses, and to examine its large knowledge base. Control of the interaction, however, rests completely with the user.

4.3 Feedback in Therapy

Another general model suggested itself in our work on a program for advising physicians on the appropriate use of the heart drug digitalis, given to patients with certain cardiac arrhythmias and heart failure [7]. The early diagnostic programs (like MYCIN, PIP and INTERNIST-I) all considered their task to be a one-shot consultation. For therapy, the most important ability is not to make the right one-time suggestion but to analyze what is happening to the patient over a longer period and to adjust therapy to best meet the treatment's goals. Figure 14 shows a schematic diagram of this feedback loop. We begin with an initial description of the patient's situation, recorded in the *patient-specific model* (PSM). This model is used repeatedly as the source of information about what are the important factors to consider at any point, what has actually happened to the patient, the program's interpretation of those events, and the program's plans and current concerns. The model is repeatedly updated as therapy proceeds, both when new facts are determined and when new conclusions are reached. This record is intended to parallel the *problem-oriented medical record*, suggested by Weed [31] as the appropriate organization for keeping hospital medical records in general.

Control flows clockwise around the loop. Each treatment cycle begins by assessing the patient's condition (as represented by the PSM). That assessment is stored in the PSM, and forms the basis for the program's next step, to formulate and issue its treatment recommendations. These recommendations and the program's expectations of what will happen in the course of therapy are also recorded in the PSM, for later comparison against what actually takes place. Of course no user is obliged to follow these recommendations, so after an appropriate time has passed (based either on the program's estimate of when some event of interest should have happened or the autonomous request of the user for a further evaluation) the program determines what actions were actually taken and what consequences followed. By comparing what happened against the expectations stored in its model, the program can quickly assess whether therapy is proceeding appropriately or needs to be significantly modified or abandoned. This program combined a knowledge of the pharmacokinetics of the various preparations of digitalis with a set of expert-provided heuristics about how to balance the effectiveness vs. the risk of side-effects of therapy. It reproduced quite well the treatment style of the expert on which it was patterned, and although its conservative style was somewhat controversial with other experts, there were indications that it was less liable than human physicians to make life-threatening decisions by overlooking important factors in the therapy plan [29].

5 State of the Art in the Late 1980's

The good news is that today the application of 1970's technology—made far more attractive by 1980's additions of graphics, smart user interfaces and ubiquitous availability on personal computers—is good enough to solve many interesting, worthwhile problems.

Ed Feigenbaum and his colleagues, in their recent book *The Rise of the Expert Company* [6], report that there are now on the order of 1500 expert system applications in actual use, with a few thousand more in prototype or field testing. These systems are being used for a number of distinct reasons, including assuring that consistent and thorough consideration is given to all relevant factors in decision making, capturing and distributing rare or vanishing expertise, and enabling more customized products and services to be assembled in response to the needs of customers. Despite the wide variety of reasons for using expert systems and the broad range of their application, a small number of central tasks come up over and over again:

diagnosis: Much as in the early systems described above, the problem is to determine an appropriate explanation for some (abnormal) constellation of observables. The task is often computing a short-term response to abnormal or dangerous conditions, for example in a chemical plant or power station, in quality control monitoring, etc.

selection: Often the central task is to select an appropriate part, the right regulation to apply, the most apt form letter. Usually there are many possibilities that satisfy some set of requirements. First one must weed out the ones that do not, then apply some optimizing (or alternatively, *satisficing*, in Simon's terminology) criterion to choose one. Often, such systems are a means of formalizing corporate rules and regulations, to get consistent and explainable behavior across the organization. Examples include insurance, credit authorization, personnel, etc.

configuration and planning: Configuration tasks are ones in which either a partial design must be completed in routine ways or a complete design checked and adjusted to meet certain constraints. A well-known example is the XCON system of Digital Equipment Corp., which is used to check the configuration of all orders for Digital computers. Complexity in such tasks arises principally from interactions among the various interdependent choices that must be made. Scheduling, which is in a sense the configuration of a set of time-dependent activities, has a similar character.

Feigenbaum points out that, perhaps contrary to initial expectations, if we classify projects as elephants, buffalos or rabbits, depending on their size and scope, there are very few elephants, a small number of buffalos and a large preponderance of rabbits. Thus, in fact, it is often the simplest of the system-building ideas from the research lab that have been packaged into convenient tools for the end-user and are now being used to create quite small applications that automate, check on or augment the capabilities of the worker who developed the system for himself. Though the continuing investment in Digital's XCON is of the order of several million dollars per year and the estimated saving resulting from it is tens of millions per year, most systems that have been built require an investment of no more than a few tens of thousands of dollars (a few man-months of effort) and yield returns that pay back that investment in well under a year. It seems that universally, when a project succeeds at all, its benefit is at least an order of magnitude greater than its cost.

What remains to be done, then? Why haven't we simply declared the field a success, turned it over to commercial interests and moved on?

5.1 Fragility of Encoded Knowledge

To illustrate some of the remaining problems, let me cite an anecdote from the late 1970's, in which one of my students decided to impress Prof. Marvin Minsky with a developmental version of PIP. Marvin, being somewhat of a joker, responded to the program's request for an initial complaint by typing the word "sick." Much to everyone's surprise, the program's immediate comeback was "Has he had his teeth cleaned lately?" The student's puzzlement only grew when he used the program's explanation capabilities to trace back through its logic to see what strange and unlikely path it followed to lead it to this unexpected question.

Alas, though this is a program about the diagnosis of kidney disease, it has no knowledge of the word "sick." Provided with a spelling checker to help ease problems of users mis-typing medical terms, the program happily accepted the word "sick" and turned it into "sigmoidoscopy," apparently its best match at correcting the mis-spelled word. Sigmoidoscopy happens to be an unpleasant procedure in which a tube is inserted from the bottom up in your intestinal tract to see if there are abnormalities indicating possible disease. The program's "logic" then proceeded roughly as follows:

```
IF  1) the infection is meningitis
    2) the subtype of meningitis is bacterial
    3) only circumstantial evidence is available
    4) the patient is at least 17 years old
    5) the patient is an alcoholic

THEN
    There is suggestive evidence (.7) that diplococcus-pneumoniae is an
       organism causing the meningitis.
```

Figure 15: A MYCIN rule that confounds several distinct issues, including diagnostic strategy, default reasoning, and social conventions about whether to ask if children are alcoholics.

1. Sigmoidoscopy can introduce bacteria into the blood stream (if the tube scrapes the lining of the intestine) and might thus lead to bacteremia.

2. Bacteremia having been suggested as an hypothesis worthy of pursuit, the program then tried to find other evidence that it may be present.

3. Dental cleaning in the recent past is commonly observed in patients with bacteremia—in fact, it is another way of introducing bacteria into the blood stream—and the program, knowing very little of interest about the patient at this point, wondered if it could thus find evidence to help confirm the only hypothesis it had been able to generate.

Of course its reaction is inappropriate, and there were a number of technical corrections that we could impose to eliminate this particular unanticipated bad behavior. For example, the direction of causality needed to be clearly marked in associations such as those between dental cleaning and bacteremia. Though in general seeking additional consequences of a condition is a good way to help establish its presence, seeking additional possible causes of a condition for which a good possible cause is already established is not reasonable. Minsky's conclusion from this episode, which can hardly be dismissed, is that the real problem is that the program has no *common sense*. According to his view, independent of its particular expert medical knowledge, it should be able to determine that the above behavior violates some common understanding that all people, not only physicians, share.

In a similar vein, Clancey discovered that although the knowledge base of MYCIN was adequate to lead to good diagnostic performance, trying to use its knowledge to help teach medicine illuminated many pitfalls. Consider, for example, the rule in Figure 15. Among the five premises, there appear at least the following component bits of knowledge:

1. Bacterial meningitis in an alcoholic is often caused by diplococcus-pneumoniae.

2. Don't rely on this if you know better.

3. Assume that children are not alcoholics (or at least have not been long enough for the purpose of the first relationship to hold).

The existing rule is perfectly adequate to cause MYCIN to behave as if it understood these distinct facts. In reality, however, it does not. When trying to build a teaching program from such jumbled knowledge, we can easily be misled into the sorts of common-sense-violating behavior we encountered above. For example, one possible reading of the rule is that being young somehow protects you from the observed association that alcoholism makes diplococcus-pneumoniae meningitis more likely.

Some, such as Minsky, have argued that problems of this sort are endemic to the enterprise of trying to build expert systems, and that only a set of fundamental advances in our ability to understand and model common-sense reasoning has any hope of overcoming such difficulties [20]. In short, this argument holds that one cannot build a robust expert without first having built a robust generalist. Thus, what prevents human doctors (for example) from making silly mistakes of the sort illustrated here is not great expertise but a lot of common experience acquired more through childhood and adolescence than through postgraduate training. Others believe that a more conventional engineering approach will yield significant advances. According to this view, what we think of as common sense is really just the accumulation of a vast stock of knowledge, orders of magnitude greater than what has thus far been encoded in any expert system [14]. Thus, if only we could catalog and encode such a body of knowledge, one could build new programs not "from scratch" but from that substantial base. The task of collecting this knowledge base is daunting. It involves not only the identification of an encyclopedic volume of information but also the development of an appropriate formalism to encode it. Nevertheless, such ambitious tasks are now underway in at least two groups [4, 14].

5.2 Complexity in the Real World

A second, and at least as serious, difficulty with the current state of the art is that many of the methods we can now apply work very well on relatively straightforward problems but fail on more complex ones. Consider the problem of medical diagnosis as we increase its complexity in the following scenarios:

1. Suppose we could guarantee that any patient was either healthy or had exactly one disease. In this ideal world (for the diagnostician), the process of diagnosis is relatively easy. Every observation may revise our relative degree of belief in every hypothesis, but eventually a single hypothesis must explain all the known data.

2. In a slightly more complex world, suppose a patient could have more than one disease, but we know that no two diseases ever effect each other or ever share common symptoms. In this case, the bookkeeping required for diagnosis becomes more complex, and we can no longer use arguments against one disease as arguments for another.

3. In the real world, patients may have not only multiple disorders but also those disorders may interact strongly. This raises serious problems of how to allocate "credit" for an observation to any of a set of diseases that may cause it, how to deal with issues of "partial credit," where no disease by itself may explain an observation but any of a number of sets of diseases may combine to do so. Therapeutic interventions often share the character of such interacting diseases—after all, the therapy is intended to affect the manifestations of the disease against which it is targeted, and so the patient's

actual condition will be some net result of the effects of the disease and effects of the therapy.

Clearly, the complexity of the diagnostic task increases greatly as we loosen the constraining assumptions about the world. Among the early diagnostic programs described above, MYCIN and PIP don't deal well with any situation more complex than scenario 1. INTERNIST-I provides the additional machinery to tackle some of the complexity of scenario 2.

Sometimes the specific goals of a project help ameliorate deficiencies in its techniques. For example, MYCIN had difficulty differentiating between the case of a single infection by an organism whose identity was in doubt and a multiple infection in which each of two or more infecting organisms are simultaneously present. Fortunately, in dealing with bacterial infections, the risks of therapy are normally much smaller than the risks posed by an untreated organism. As a result, it happens that the appropriate response to either of these situations is to treat for all of the possible identities of all suspected organisms. If it turns out that only a single infection is present, treating for all will surely cover the right one, whereas if there are multiple infections, treating for all will also cover all the right ones. Thus, by a fortunate truth about the domain, this potential weakness in the program doesn't matter. Alas, such strokes of good luck do not generalize well. For example, if one of the possible infecting agents requires treatment with a drug with possible serious morbidities, then a human expert would put a much greater emphasis on distinguishing the two situations posed here.

Among the generalizable models of expert reasoning that are understood thoroughly enough to have reached the commercial marketplace, very few provide facilities that go beyond the single-problem scenario we have outlined. And virtually none help with problems of rampant interaction suggested in the third scenario. This one is truly difficult, as we have come to realize in some of our work on the diagnosis and therapy of acid/base and electrolyte imbalances. Consider the following medical facts:

- Diarrhea causes bicarbonate (alkali) loss (*lowered pH*).
- Vomiting causes acid loss (*increased pH*).

As a result, a patient who has diarrhea and is vomiting may in fact exhibit a *normal* pH. To capture this in a MYCIN-like system, we would need some equivalent of

```
IF   the patient has a normal pH
THEN There is suggestive evidence that the patient is suffering
        from diarrhea and from vomiting.
```

which is, on the face of it, an absurd interpretation of medical knowledge. Because virtually any abnormality may be cancelled out by another under appropriate circumstances, this path leads to saying that every normal finding suggests a vast number of these cancelling combinations of diseases.

Clearly, something more fundamentally sound is needed.

5.3 Sources of Better Models

An expert physician knows a broad range of material. He or she has learned the "basic science" of medicine, including college and then medical school courses in biochemistry, physiology, etc. Medical school teaches the specifics of medicine: pathophysiology, pathology, genetics, etc. The expert has also learned though a process of clerkships, internship and residency the field's state of the art or practice: the available tests, treatments and procedures. In addition, experience teaches much of value: how to apply one's knowledge most effectively, what occurs commonly (learning not to guess "zebra" when hearing hoofbeats, in the parlance of the field), local wisdom as distilled by colleagues about the preferred ways of approaching various problems, etc.

Someone embarking on research in the field of medical decision-making might in fact be tempted to approach its problems from the traditional scientific front: Build appropriate models of what is known, then apply that knowledge to the individual patient to answer any questions. For example, decades of research have led to fairly specific models of some aspects of human physiology, such as the control machinery of the circulatory system [9]. Why not plan therapy for heart disease, then, by adjusting such a model to the patient at hand and simply predicting the model's response to all therapies under consideration? Alas, the typical physiologic model contains hundreds of parameters, and determining their values for an individual requires much more data than could be reasonably collected. In fact, collecting those required data might not only be tedious but could well require challenging the patient's system in ways that are unacceptably risky. Yet in the previous section we have argued that ignoring true models that underlie our experiences leave us with too little knowledge to handle tough cases.

For most domains, not just for medicine, we really do not yet know how to capture the desired breadth and depth of knowledge in a computer, or how to develop programs that effectively call forth just the right parts to produce expert behavior. The following sections present short descriptions of a number of interesting ideas that have evolved and been explored in the past few years. The first batch will be drawn from medical reasoning, whose difficulties make it the area that has demanded the most of evolving AI methods. The second batch will illustrate that similar needs and opportunities exist as well in other areas of knowledge-based systems. Each project has exploited some insight about its domain and task to make possible the use of a representation and a reasoning method that is intermediate between the simple early systems we have seen and the demanding formal models of scientific investigation. The common thread seems to be finding the right level of detail and precision, and ignoring just those factors that can be omitted from the analysis while still getting significant value from it.

6 Sophistication in Medical Reasoning

We examine a handful of research projects that appear to point in appropriate directions for progress in AI applied to medical reasoning. The first addresses the problem we pointed out earlier: how to deal with disease hypotheses that interact strongly. The second concerns the use of qualitative versions of physiological models, to derive results that are useful clinically without having to resort to massive data collection. The third introduces a fundamental advance in thinking about probabilistic relationships.

Unfortunately, each of these projects exists in isolation of the others, and it is a difficult but important research goal to formulate ways of achieving simultaneously in a single system the advances of each.

6.1 Multi-level Models in Acid/Base Diagnosis

In critiquing the inadequacies of first-generation expert systems, we mentioned the need for diagnostic programs to deal effectively with interactions among disorders, even ones that might effectively cancel out some of their abnormal effects. Ramesh Patil, in his MIT doctoral thesis [21], introduced several important notions that combine effectively. First is that, at least in domains in which some of the knowledge and observations are quantitative, the program must be able to deal explicitly with not only the presence or absence but the severity of any abnormalities. For acid/base disorders, where at least some of the models are well-known and quite precise, one may even make quantitative calculations to assure consistency in the model. For example, in a hospitalized patient with severe diarrhea, the quantity and composition of the fluid lost may have been measured and recorded, allowing an accurate assessment of the body's losses of fluid, bicarbonate, potassium, etc. Even if the quantities are not known with precision, requirements of material balance and similar simple notions can help point out unsuspected but important components of a complex disease state. For example, if the degree of observed acidemia is inconsistent with the known sources of bicarbonate loss, the program can legitimately speculate about offsetting factors. In the patient with diarrhea, to return to our earlier example, the absence of a low serum pH would suggest some other mechanism of bicarbonate gain or hydrogen ion loss. Thus, the program might investigate whether the patient has also suffered from vomiting to a sufficient extent that the loss of the expected additional hydrogen ion can be explained.[6] In this way, a normal pH can indeed serve as supporting evidence for the diarrhea-and-vomiting scenario, but without the need to encode that and all other such offsetting pairs in the program's associational database.

The second major contribution of this work was to introduce the notion of multi-level models. As suggested in Figure 16, it is often very useful to be able to consider a case at a number of interrelated levels of detail, ranging from the experiential clinical associations represented by the top level to the pathophysiological details represented at the lowest. The vertical connections, which ultimately tie the clinical relationships to the physiological mechanisms that account for them (if known) allow information to flow in both directions. Upward, the clinical significance of laboratory data can be assessed. Downward, clinical impressions can constrain the interpretations that may be derived by the detailed models. In principle, this sort of organization also allows drawing the relationships between possibly-related observations at the most appropriate level where knowledge of those relationships exists. In this particular program, however, we were unable to exploit this capability, because the program always tended to try to elaborate every model as deeply as possible. As a result, even straightforward cases in which the associations were perfectly adequate to arrive at the correct answer would become subject to a deep, detailed and slow analysis.

The combination of its abilities to deal with models at multiple levels of detail and to investigate components of abnormal values gave ABEL a powerful basis not only for reaching sophisticated diagnostic conclusions but also for planning an effective strategy of diagnostic

[6]The fluid lost in vomiting is normally rich in acid.

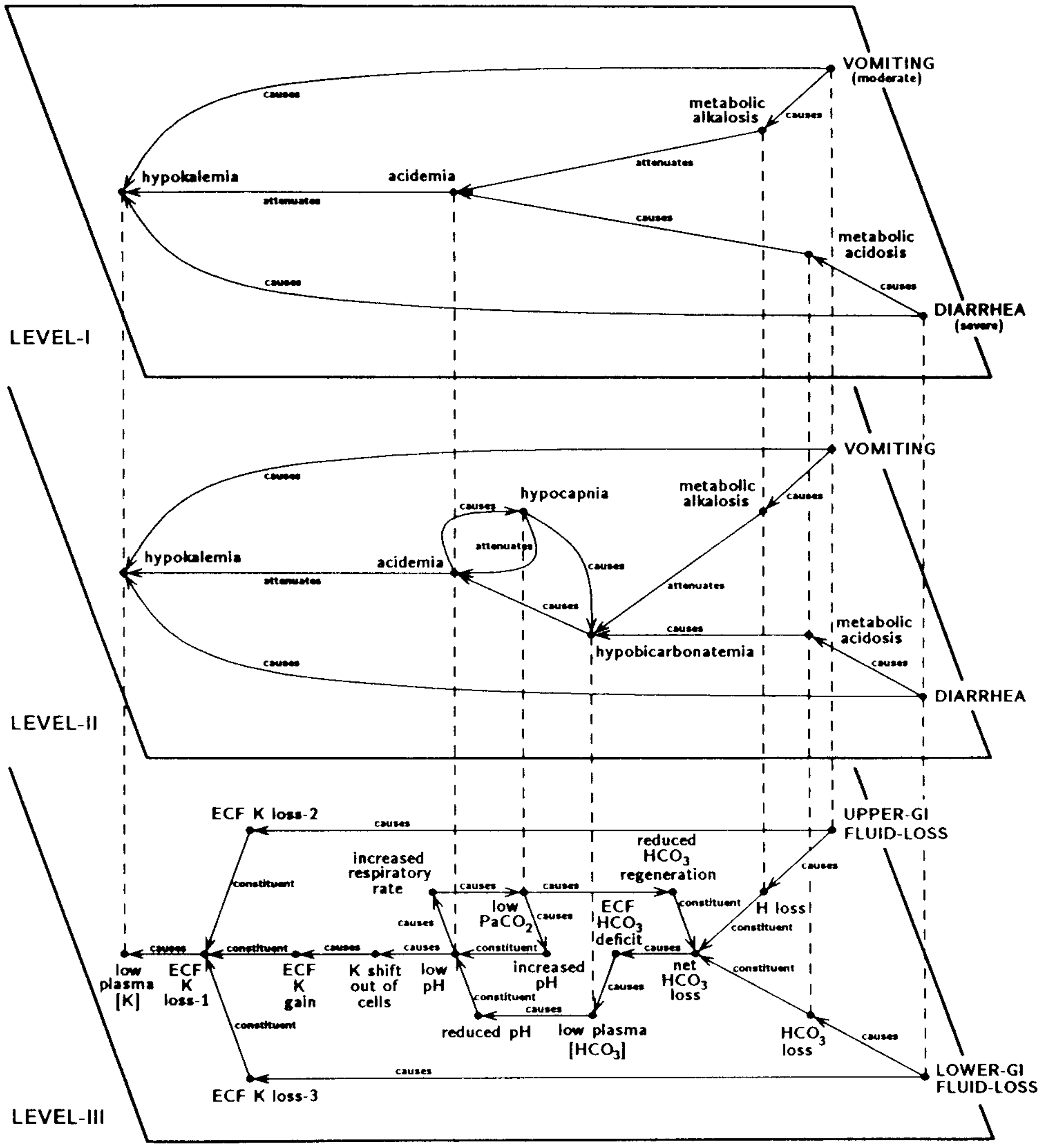


Figure 16: ABEL's multi-level model of a case with diarrhea and vomiting. Level I depicts clinical associations, level II an intermediate level of detail, and level III the physiological mechanisms involved in the case.

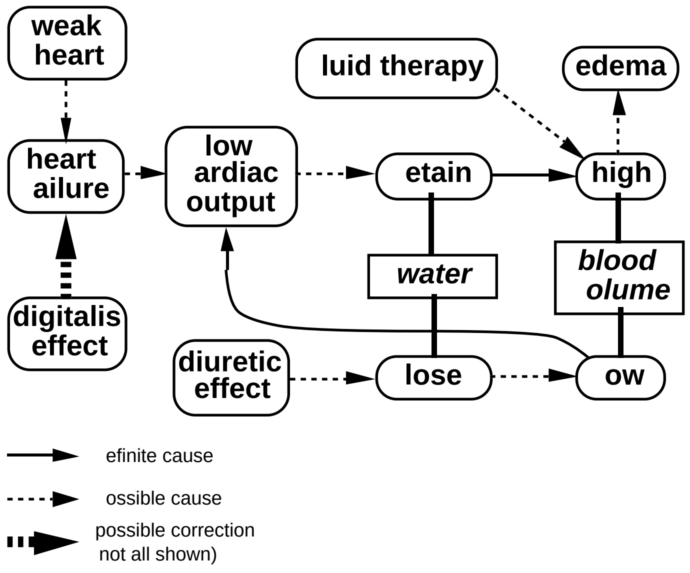


Figure 17: Heart-Failure Program's simplified model of fluid balance, indicating the effects of a weak heart, use of the drug digitalis, fluid therapy and diuretics on blood volume and edema (swelling in the tissues caused by an accumulation of fluid). Each arc is also associated with a typical delay, which is not shown here.

testing and for producing good graphical and English-language explanations of its reasoning. Its principal limitations resided in the rather special nature of its domain—one of the best in medicine for having available deep quantitative models—and in the complexity of the program and data that made it difficult to extend even to related fields, much less to ones from other domains.

6.2 Qualitative Physiological Models

Often, to understand the behavior of a complex system one must abstract away from the particular details to a more qualitative view. The same observation underlies the enormous preference of traditional scientists and engineers for closed-form solutions to complex systems. After all, it is much easier to think about what a system does given a description of that behavior in a simple equation than to pore over dozens or hundreds of simulation runs, trying to comprehend it. Unfortunately, very few systems of real-world complexity and interest have classical closed-form solutions, so the name of the game is to find the right approximations that allow compact descriptions of the system that are nevertheless reasonably accurate, at least to the degree required for the task.

AI researchers have introduced a large number of related ideas and systems, going under names such as qualitative physics, qualitative simulation, qualitative process theory, etc. Despite important variations, each of these represents quantities of interest, including levels, rates of change (and sometimes higher derivatives) not by numbers but by more abstract quantities. In some cases, these can specify only whether a value is positive, zero or negative.

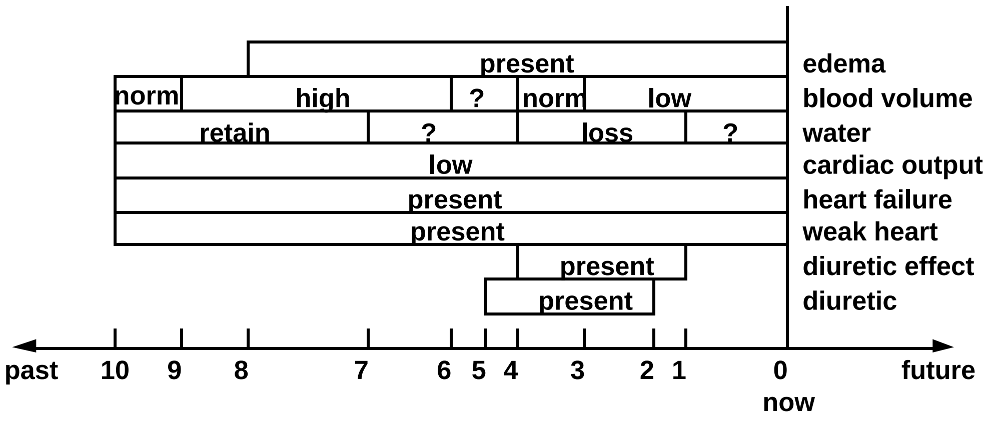


Figure 18: Postdiction for a case of heart failure with edema and low blood volume. The time line stretching back into the past shows significant events as numbered times. The specific time of these is not known, only their relative order. Each line in the diagram represents the state of a variable of interest at times in the past. The history of states shown is consistent with a case in which (at the present time) the patient has edema in the face of a low blood volume.

Other variants allow for a small number of "landmark" values that partition the space of quantities into equivalence classes. Given these, any value may be described as belonging to one of the intervals between landmarks (and usually ∞ and $-\infty$ are considered landmarks).

William Long, at MIT, developed an interesting model of this general character to help reconstruct a scenario of past events in order to explain the current presence of an apparently contradictory situation [16]. Consider a heart failure patient with low blood volume and edema (an accumulation of fluids in the tissues). This situation, on the face of it, appears contradictory, because low blood volume suggests an overall loss of fluid from the body, whereas edema suggests its gain. Such a situation is, in fact, an implausible steady state, but can arise at times transiently because some of the body's responses are virtually instantaneous, whereas others are manifested only over long periods of time. To resolve this apparent contradiction, we need a system that can recognize and exploit the different speeds with which various physiological mechanisms react.

To understand this case and similar ones, we need a qualitative model of how the underlying disease, its effects and possible therapies interact, and of the typical time course of these effects. Figure 17 shows a highly simplified model of the consequences of heart failure. With each causal relationship there is also an indication of the typical time over which it acts. From this information it is possible for us (and for the program) to reconstruct a plausible history of how the patient arrived at the current state. In Figure 18, we can trace the following chain of events back from the present, relying on the model of Figure 17 to justify each intermediate conclusion:

1. According to the model, the only way to reach a low blood volume is to have been losing water for some time in the past, shown here in the time interval from points 4 to 1. We do not know if water loss is continuing now. (Note that these time points are

ordinal, but have no specific durations implied.)

2. The only way to lose water, in this model, is to be acted on by a diuretic. Because diuretics act with a small time delay, we conclude that the diuretic effect was present between times 5 and 2, which precede the beginning and end points of the period of water loss, respectively.

3. Because volumes can change only continuously, blood volume must have been normal before it became low, with the transition occurring at some time 3, which must fall during the period of water loss.

4. According to the model, the only way to have edema is to have had a high blood volume for some time in the past, here shown in the interval 9 to 6.

5. The only way to develop a high blood volume, according to the model, is to retain water, here shown in the interval 10 to 7.

   ...

In this way, we can reconstruct a plausible history. The process, which might be called *postdiction*, is essentially the dual of qualitative simulation, or prediction, where we try to describe possible future states of a system from knowing its present and its laws. In the case of postdiction, we also benefit from strong constraints imposed by any real facts we know about the actual past. For example, knowing the time of diuretic administration gives a fix on time points 2 and 5, and thereby constrains all the others.

The power of methods such as this is great. Note that it permits a conceptual shift in what we mean by the notion of diagnosis. We no longer ask for the name of a disease, but for a temporally and causally-connected historical reconstruction. If combined with the ability to reason about the quantitative aggregation of effects, as suggested above, this can form the basis for a very strong diagnostic reasoner.

6.3 Qualitative Probabilistic Modeling

We noted in our introductory discussion of expertise that virtually nothing of human knowledge is truly certain. Then, we argued that despite this uncertainty, we often act as if we believed things with certainty, perhaps buttressing our reasoning with special mechanisms to help get out of trouble when the inevitable false assumption actually leads to a blunder.

Though this approach appears quite workable for many areas of human competence, there are others in which a far more explicit devotion to the study of probabilistic relationships is required. For example, there are many areas of medicine in which the theoretical underpinnings are so weak that most real decisions have to be made based on the carefully aggregated experience of past cases. In trying to assess the risks posed by various drugs or environmental conditions, in attempting of optimize the empirical treatment of cancer by an ever-growing pharmacopeia of noxious agents, we may understand relatively little of just how or why an agent works, but we can nevertheless extrapolate from past experiences by treating them as a statistical sample, and assume that a new patient is drawn from the same population and therefore can look forward to (statistically) the same range of outcomes.

AI research has traditionally eschewed the straightforward application of these ideas because of a sense that statistics tend to sweep the interesting differences under the rug rather than exposing them to explicit scrutiny. Thus, when a failure is noted in a statistically-based decision-making program—e.g., a diagnostician mis-diagnoses a known case—the statistical approach may be content to dismiss this error as the rare but expectable error case; it is, after all, expected that only 95% or so of the cases will fall exactly within the model. By contrast, the AI approach would be to try to examine in detail what led to the wrong conclusion, and to learn from it some distinguishing feature that might help avoid this error in the future. In brief, AI researchers often act as if they believed that noise is not random but is simply the result of a deeper, yet discoverable process.

The other major argument against probabilistic reasoning in AI has been the observation that such reasoners tend to treat probabilistic relationships in very uniform ways, either imposing unrealistically strong assumptions such as conditional independence across the board, or requiring the collection of huge databases to provide adequate data to assess all the dependencies. Recent advances in the methods for analyzing probabilistic networks of dependencies [23] have dramatically improved our abilities to take true probabilistic models into account in AI reasoning programs. Nevertheless, many researchers sense that such models are highly sensitive to the particular distributions of probabilities that stock their models. Indeed, a probabilistic decision model tuned to work very well at deciding which patients are suffering from appendicitis at one institution falters when moved to a different setting, and regains its acumen only when re-trained on a large volume of locally-gathered data [2].

Just as we noted in the case of qualitative models of physical or physiological systems, there is a strong argument to be made for developing qualitative models of probabilistic relationships. An example of such a qualitative statement is that the presence of the heart drug digitalis in a patient's blood increases the automaticity of the heart (the tendency of muscle fibers to fire spontaneously). Though we are uncommitted about the degree to which we will find this increase, the general statement can be interpreted to mean that the higher the level of the drug, the more likely is the spontaneous firing. At this level of generality, the relation is virtually a corollary of cardiac physiology, and is likely to be very robust to changes in the subject population. Thus, though the strength of this relationship may vary from one group of patients to another, the sign of the relation is almost sure to remain the same.

Figure 19 shows part of Michael Wellman's model of the interactions of digitalis, serum calcium and serum potassium on the control of the heart [32]. Digitalis is a drug often used to treat (among other things) tachycardia, a high heart rate. The benefits of treatment are summarized in the top path in the graph: digitalis tends to decrease the conductivity of the heart. Increased conductivity leads to an increased heart rate, therefore a decrease in conductivity tends to lower the heart rate. Because, in this instance, a higher heart rate is bad for the patient (leads to decreased value), the lower heart rate expected from decreased conductivity is beneficial. The lower path in the figure shows that increased digitalis, increased serum calcium and decreased serum potassium all increase automaticity, which increases the risk of ventricular fibrillation, an often-fatal condition that clearly decreases value.

In the description given so far, we have said nothing about the magnitudes of the expected effects, only their directions. It is not surprising, therefore, that when we aggregate this

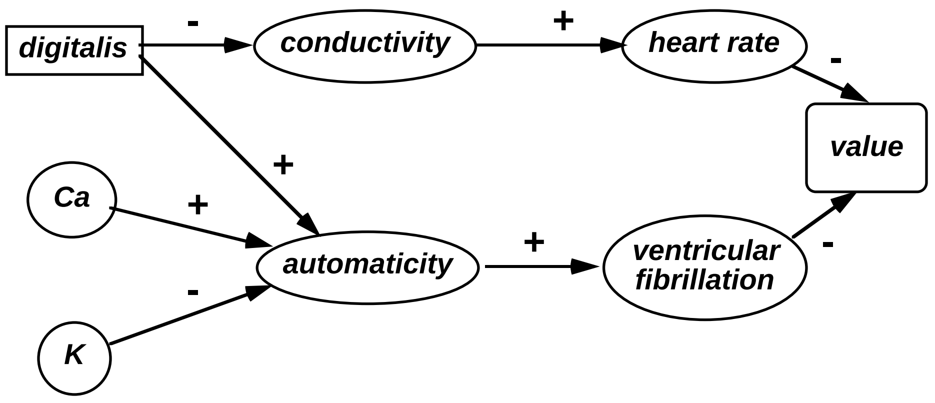


Figure 19: Effects of digitalis and serum calcium and potassium on the heart and the patient. Digitalis has a beneficial effect, indicated by the top path to *value*: by decreasing conductivity, it leads to a lower heart rate, which is beneficial for the patients under consideration. The bottom path shows a possibly-fatal mechanism of toxicity: increasing digitalis increases automaticity, which in turn increases the risk of ventricular fibrillation, which is very highly undesirable. High levels of calcium (Ca) and low levels of potassium (K) in the blood also contribute to increased automaticity.

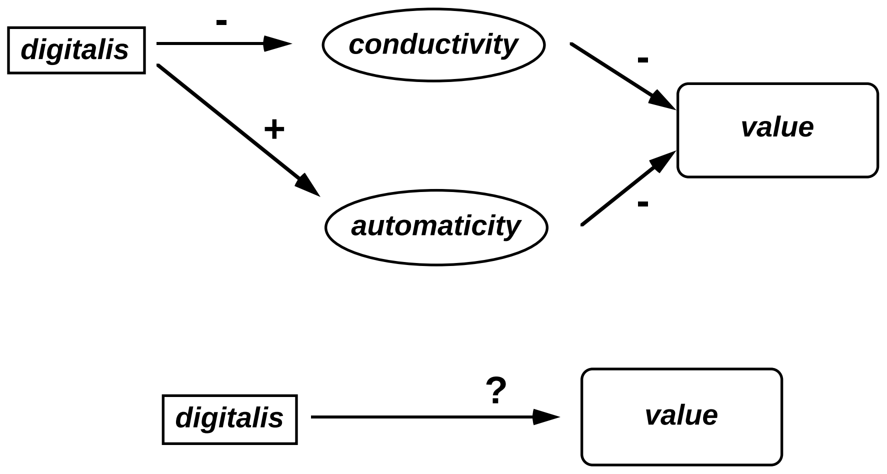


Figure 20: In this model, there is not enough information to tell the aggregate effect of giving digitalis. The qualitative graph-reduction operations defined in [32] lead to an ambiguous result, showing that there is not enough information, in general, to tell whether the benefits or risks of the drug are greater.

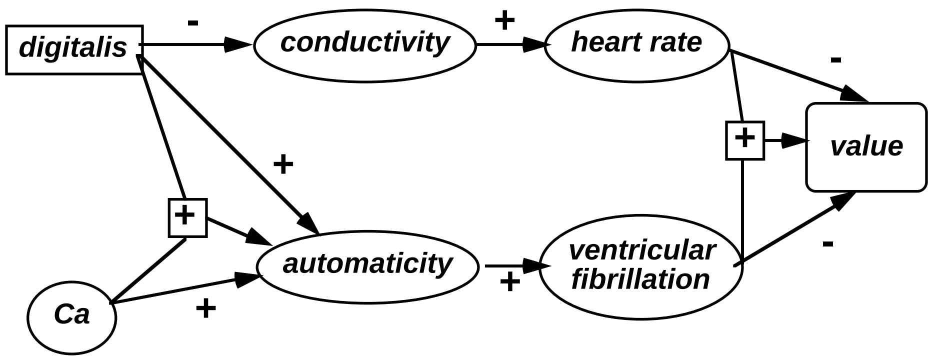


Figure 21: Introducing synergy into the model. Positive synergy between *dig* and *Ca* on *automaticity* indicates that the likelihood of digitalis causing automaticity is greater in the presence of higher levels of serum calcium. Positive synergy between the negative effects of heart rate and ventricular fibrillation serves to model indifference to heart rate when ventricular fibrillation is present.

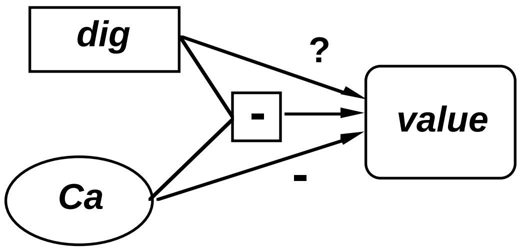


Figure 22: Though the effect of giving digitalis is still uncertain, synergy in the model implies that the optimal digitalis dose must be a decreasing function of serum calcium.

information (in Figure 20) to ask "is giving digitalis a good idea?" the system is unable to tell because it cannot determine whether the beneficial or harmful effects of the drug will predominate.

However, it is possible to get useful information out of such a model without knowing quantitative relations if we add more qualitative knowledge about the joint effects of drugs and existing conditions. For example, there is a synergistic interaction between the effects of digitalis and calcium on automaticity, as shown in Figure 21. This means that a given change in digitalis has a greater effect on automaticity when serum calcium is high than when it is low. Applying Wellman's calculus for drawing implications from such qualitative models, we arrive at the diagram of Figure 22: though we still cannot tell whether giving more digitalis is a good idea or not, we can tell that the optimum dose of digitalis is a decreasing function of serum calcium. This is in fact a result we had explicitly encoded from experience in the digitalis advisor, but the new qualitative probabilistic reasoning methods allow us to derive it from very general and robust principles.

7 Model-Based Reasoning

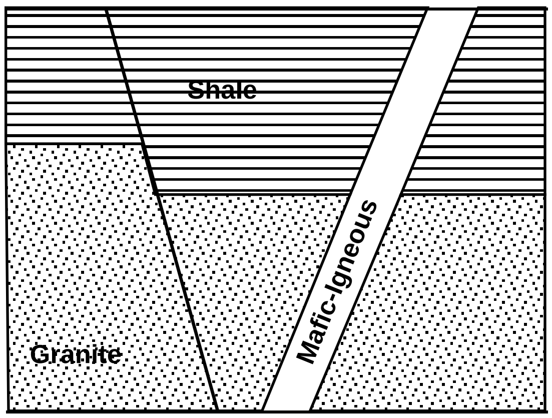
The above examples have been chosen from medical applications, which have indeed often inspired the development of the relevant ideas. Medicine is by no means the only field, however, that requires reasoning more deeply about systems. In cases where simple experiential knowledge turns out to be inadequate, we need to build suitably-abstract models of the domain and then reason about those models to help answer difficult questions.

The fundamental notion of a model, in expert systems as in traditional fields like bridge design, is a smaller, simpler entity than the actual system that nevertheless exhibits many of the properties of interest in the real system. If the model is well-designed, simple reasoning about the model may be able to answer difficult questions of genuine interest about the actual system. AI systems often emphasize the desire to model explicitly only the structural components of a system and then automatically to derive the behavior of the system. When the behaviors and interactions of the component parts are well-enough understood, this is a very powerful method because it means that any system consisting of such parts can be successfully modeled merely by writing down the parts and their connections, and the remainder of the analysis of the system can take place automatically. In practice, issues such as lack of precise knowledge and the combinatorics of modeling interesting systems with a very large number of parts conspire against this simple view and force on the model creator a demand for careful thought about just what to include in and exclude from the model.

In this section, we examine briefly two recent theses in model-based reasoning from Randy Davis' group in the AI Lab, one concerning geological reasoning, the other reasoning about the behavior of complex electrical circuits.

7.1 Using Models for Geological Interpretation

Reid Simmons' chosen problem was to give a geological account for an observed cross-section through the earth, such as that shown in Figure 23. A plausible interpretation of this picture corresponds to the following tale: Shale was deposited on the site. Granite then intruded



Interpretation:
1 Deposition1(Rock1,Shale)
2 Batholithic-Intrusion2(Rock2, Granite)
3 Uplift3(Uamount1)
4 Faulting(Fault1)
5 Dike-Intrusion5(Ign1, Mafic-Igneous)
6 Erosion6(Elevel1)

Figure 23: Gordius' diagram of a geological cross-section, and its interpretation. The diagram shows a cross-section at the present time, and the interpretation gives a plausible course of events that could have resulted in the given cross-section.

under it. After the region was uplifted, the fault toward the left of the diagram occurred. Then a mafic-igneous rock intruded across the right, and finally erosion leveled the top. The gordius program generates just such interpretations [25].

In a sense, the problem here is analogous to Long's attempt to reconstruct the history of a complex medical process. In geology, added complexity derives from the geometric complexity of the diagram, the very large number of (combinations of) mechanisms that could have accounted for almost any part of the diagram, and the fact that we almost never have real data about what was actually true in the past (as might be the case in a carefully-tracked medical case).

Gordius' approach is summarized as a sequence of three steps, the first two of which are analogous to the strategy developed for dendral. First, features of the diagram are used to generate hypotheses, which are constrained by relations in the diagram to fit together only in a few acceptable ways. Second, a qualitative simulation checks whether the hypothesis could in fact generate the currently-observed diagram. Finally, a debugging process is used to adjust the hypothesis to eliminate discrepancies pointed out by the simulator. A hypothesis is repeatedly simulated and debugged until its consequences are consistent with observations.

7.2 Temporal Abstraction for Reasoning about Complex Circuits

To drive home the importance of choosing the correct abstractions for reasoning about complex systems, consider the (cramped) portion of the circuit diagram of a Symbolics 3600 video controller board shown in Figure 24. In the natural representation of digital circuits, where we may view each logic element at any time as being either on or off, almost any small part of this circuit is sufficiently complex that it exceeds the reach of our meager analytical tools . To make matters worse, many circuit parts have internal state, and a number of the wires in this diagram are in fact buses carrying multiple, interrelated signals. Simply simulating the behavior of such a circuit given complete knowledge of the behavioral rules of its parts and its initial state is a complex and time-consuming task. Trying to figure out what

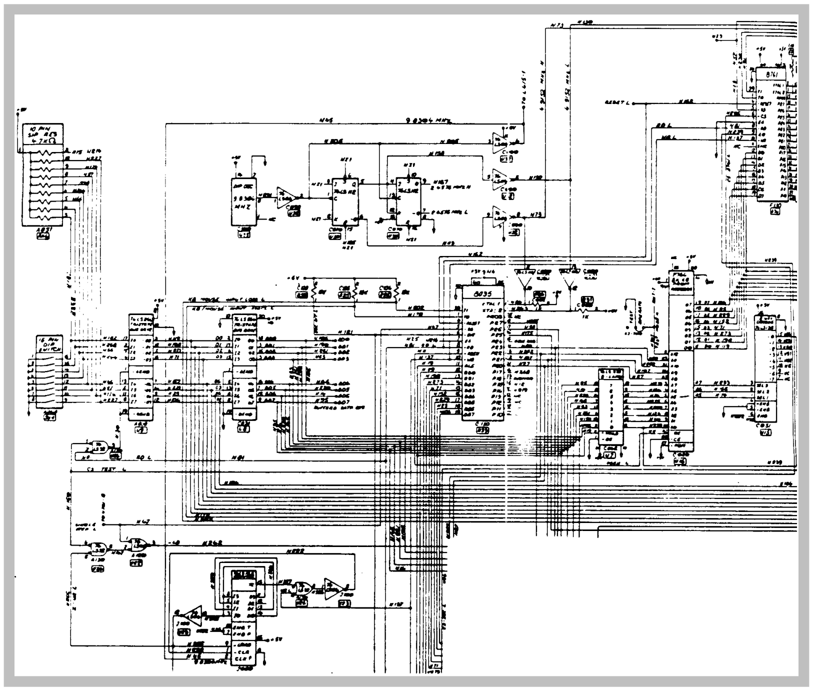


Figure 24: Part of the circuit diagram of a Symbolics 3600 video controller.

is wrong with it by postulating defects and then simulating until the observed abnormalities are reproduced seems hopeless—at least at the detailed bit-by-bit circuit level.[7]

Walter Hamscher actually spent a summer repairing faulty versions of this board and learning how technicians approached the problem. He discovered [11] that they use a rich language of temporal abstractions to describe behavior, including concepts such as: stay-same, change, duration, count, sequence, cycle, frequency, sampling, etc., and their compositions. Once the circuit and its parts are understood in such abstract terms, relatively simple diagnostic rules suffice to help direct suspicion to the right candidates. For example, a *very* simple diagnostic rule might be:

> A circuit component that is supposed to be a clock, but whose output is not changing, must be faulty.

Here we are not speaking of individual gate transitions, but rather of much more appropriate aggregate behavior. The notion of a clock embodies a sequence of regular transitions, with a certain frequency. "Not changing" is a simple, easily tested aggregate behavior. When described and approached in such terms, even the apparent complexity of Figure 24 seems approachable.

[7]Notice that the argument here is very much the one we made against physiological simulation to diagnose human disease. Randy Davis and I sometimes joke, in fact, that our two domains—medicine and circuits—are not so different. The main differences are that in his, he can at least assume a circuit diagram that God has not chosen to publish for people; whereas in mine, I only have to deal with one basic model.

8 Software Methodology

The successful exploitation even of the somewhat limited means at our current disposal has some interesting lessons for the field of software methodology. In contrast to the rather formal specification-implementation-verification-based methods advocated by many (including John Guttag in this volume), the expert systems field places a great emphasis on the incremental interweaving of specification, design, implementation and testing [26]. The fundamental driving force for this radically different orientation is that in building expert systems it is quite typical that the developers do not fully understand what their system is really supposed to do until they have worked seriously with experts from the domain, implemented various preliminary ideas, revised their goals based on test cases, etc. How could one specify *a priori* what it means to build a system for providing therapeutic management assistance to doctors treating patients with heart failure? What is the "correct" specification of a geological interpretation system? Initially neither the expert nor the system builder may have a clear idea. Further, such a clear idea may not be able to emerge until the developers have had a chance to assess just what is technically feasible (within the time and resource constraints of the project) or how useful it actually is to get the sort of interpretation or advice that was initially the goal. It is mostly through such experience that new task requirements or new modes of use are worked out.

The fact that in a very real sense expert system developers "don't know what they're doing" means that the greatest emphasis must be placed on high-quality development environments. Such environments include support for rapid prototyping, incremental debugging, and the ability to postpone commitments. This often means systems that provide both compilation and interpretation, generic operators, "object-oriented" programming, all in the interest of simplifying the process of changing one's mind. These environments may also take the form of shells or toolkits that pre-package simple routine methods for solving various problems or problem components. In light of the observation that many of the systems being built are small, often built by the domain expert, these environments must also support the illusion that anyone, even "non-programmers," can program.

What is new to software engineering in the tools that currently support practical system development? Primary is the adoption of a "very high level language" viewpoint—direct support for problem-solving abstractions that are not present in conventional programming languages. Also important is support for the characteristics of expert systems (listed below), primarily the explicit representation of knowledge and built-in facilities for explanation and the examination and updating of the system's knowledge.

The final advance brought to software engineering is mainly an interesting and important matter of perception. Many expert system languages and tools appeal to the end user to program his or her ideas directly. We have suggested some of the technical reasons why this will strike many as more plausible than a similar appeal for end-users to take up COBOL or C directly. In addition, however, it appears that there is a component akin to Dumbo's magic feather[8] in these appeals. Interestingly, the motivated end-user discovers that he or she can, indeed, fly!

[8]For those without small children, in the Walt Disney classic animated film Dumbo discovers that he can fly with the aid of a "magic" feather. Upon accidentally losing the feather, his fast-talking mouse assistant has a hard time convincing him that he really can fly, feather or no.

9 Conclusion

We have argued that the embedding of large quantities of knowledge in computer programs is an excellent technique for enabling us to build sophisticated applications, called expert systems. We examined the roots of this idea, looked at some of the simplified ways in which it has reached commercial application, considered some of the problems remaining in our basic conception of how to build such systems, and looked at a handful of interesting efforts recently emerged from the research laboratories that point toward dramatically improved future capabilities.

In conclusion, I want to take a retrospective look at what characterizes knowledge-based, or expert, systems, and to reflect on the difficulties that lie ahead in adapting today's exciting research ideas to practical use.

Having looked at a number of expert systems, can we now say just what they are? After all, any system is, in a sense, an expert at whatever it was designed to do, and saying that knowledge-based systems embody knowledge can't really distinguish them from other programs—after all, how many programs are there that don't in some sense contain knowledge? What do we mean, then? Perhaps the following list of characteristics is a useful guide:

1. Expert systems typically operate in domains in which there are many possible answers to most problems. Therefore, the naive approach to problem solving, as suggested in our robot and alarm clock example, will involve search. The prime responsibility of the expert system is to apply its knowledge to eliminate or significantly reduce the magnitude of the search. Thus, expertise yields the ability to get an adequate answer without systematically exploring nearly all the possibilities.

2. Expert systems, like most AI systems, use predominantly symbolic rather than numerical computing methods. Thus, the data structures are often composed of complex interlinked structures.

3. Explicit representations of the universe of discourse typically play a causal role in the reasoning of the system. Thus, the system works by manipulating an explicit encoding of its knowledge, which is thought to be an abstraction and decomposition of the most important aspects of the real world. A corollary is that the system typically has some form of self-knowledge: it can explain what it knows explicitly, and trace out the reasoning steps that led from its observations to its conclusions.

4. The problem-solving methods are often deliberately human-like, patterned on the observed behavior of human experts. Sometimes this is because no other theories of the domain are available. In other circumstances, it is motivated by the desire to make details of the program's operation meaningful to human experts who may have to evaluate it or work with it.

5. Finally, expert systems often contain an explicit representation of their own goals. They can therefore participate in the choice of the most appropriate problem-solving methods for addressing the task at hand.

9.1 Difficult Challenges

In the late 1980's, there appear to be only a few principal challenges standing in the way of our ability to create much more sophisticated expert systems.

First is the need to conjoin the clever insights and techniques exemplified by some of the research projects above. Though each program achieves an impressive demonstration of needed capabilities, we have little idea how to combine these coherently. Thus, a program that reasons about a multi-level causal hierarchy of hypotheses while at the same time including appropriate temporal abstractions and postdiction machinery while at the same time allowing the expression of quantitative and qualitative probabilistic relationships is not currently in sight. A few years ago, our intuition was that progress in knowledge representation research would slowly lay the groundwork to achieve just this sort of integration, if not at one stroke, at least by providing powerful representation languages that would allow us to express all the relevant knowledge in a single language [28]. Given that start, we could then evolve toward systems with multiple abilities to handle all aspects of very complex domains and problems. Alas, this intuition has at least temporarily foundered on harsh experience [3, 10]—the current languages just are not up to the task.

The second difficulty is the sheer size of the knowledge base needed for a program to have a comprehensive understanding of a field. In 1976 our colleagues [22] arrived at "an upper bound of approximately one million facts as the core body of information in general internal medicine." Even to encode such a vast body of knowledge once is a daunting task. Lacking an agreed-upon universal representation language (at least one that is easier for computers to manipulate than English) means that this task may need to be undertaken repeatedly, as new ways of using it are discovered.[9] One consequence is that research in the field is slow. Between the need to develop interesting new ideas, to implement these as complex and large computer programs, and to describe a respectable chunk of knowledge in a form appropriate for the new program, five years passes easily in the life of a researcher (read "graduate student").

The problem of acquiring the required knowledge is being attacked by two routes. The frontal assault simply treats the difficulty as a very large engineering problem. Thus, such projects as Lenat's CYC [14] plan to build a very large encyclopedic knowledge base by incrementally resolving representation problems, inventing conventions as needed, and steadily accumulating a valuable collection of descriptions of aspects of the world. Similarly, though in a more limited domain, the UMLS (Unified Medical Language Systems) project attempts to develop translation methods to allow the interchange and integration of knowledge among various existing representation and indexing schemes [4]. The flanking maneuver assumes that the best way to gather the required knowledge is by developing techniques for the computer to learn it directly from interactions with human experts, direct investigation and experimentation under autonomous program control, or experience as described in large databases. Learning is currently a vibrant topic of AI research, but, just as with advanced expert systems techniques themselves, the results are more promising than achieved.

[9]In addition, in many rapidly-evolving domains such as medicine, the knowledge itself changes with dramatic speed. This means that quite aside from the technical reasons for re-encoding knowledge, portions of it may need to be continually re-encoded just to keep up to date.

9.2 The Future

Ultimately, it seems impossible to separate the long-term advance of expert systems from AI research in general. Rather simple tools available today, based on ten-year-old ideas, empower users to build valuable applications. As we succeed in integrating and simplifying today's research ideas, new generations of expert system technology will enable tomorrow's users to build much more sophisticated programs. Eventually, every interesting advance at the roots of AI research feeds directly into better experimental expert systems, and finally into applications.

References

[1] William J. Clancey. From GUIDON to NEOMYCIN to HERACLES in twenty short lessons. *AI Magazine*, 7(3):40–60, 1986.

[2] F. T. de Dombal, J. R. Staniland, and Susan E. Clamp. Geographical variation in disease presentation. *Medical Decision Making*, 1:59–69, 1981.

[3] Jon Doyle and Ramesh S. Patil. Two dogmas of knowledge representation: language restrictions, taxonomic classifications, and the utility of representation services. TM 387b, Massachusetts Institute of Technology, Laboratory for Computer Science, 545 Technology Square, Cambridge, MA, 02139, September 1989.

[4] David A. Evans and Randolph A. Miller. Final task report (task 2)—unified medical language system (UMLS) project: Initial phase in developing representations for mapping medical knowledge: INTERNIST-I/QMR, HELP, and MeSH. Technical Report CMU-LCL-87-1, Laboratory for Computational Linguistics, Carnegie Mellon University, Pittsburgh, PA, 1987.

[5] E. A. Feigenbaum, B. G. Buchanan, and J. Lederberg. On generality and problem solving: A case study using the dendral program. In B. Meltzer and D. Michie, editors, *Machine Intelligence 6*, pages 165–190. American Elsevier, New York, 1971.

[6] Edward Feigenbaum, Pamela McCorduck, and H. Penny Nii. *The Rise of the Expert Company.* Times Books, 1988.

[7] G. Anthony Gorry, Howard Silverman, and Stephen G. Pauker. Capturing clinical expertise: A computer program that considers clinical responses to digitalis. *American Journal of Medicine*, 64:452–460, March 1978.

[8] Computer-Aided Mathematics Group. *Macsyma Reference Manual.* Symbolics, Cambridge, Mass., version 12 edition, June 1986.

[9] A. Guyton, C. Jones, and T. Coleman. *Circulatory Physiology: Cardiac Output and its Regulation.* W.B. Saunders Company, Philadelphia, 1973.

[10] Ira J. Haimowitz, Ramesh S. Patil, and Peter Szolovits. Representing medical knowledge in a terminological language is difficult. In *Symposium on Computer Applications in Medical Care*, pages 101–105, 1988.

[11] Walter Charles Hamscher. *Model-Based Troubleshooting of Digital Systems.* PhD thesis, Massachusetts Institute of Technology, August 1988.

[12] David Heckerman. Probabilistic interpretations for MYCIN's certainty factors. In Laveen N. Kanal and John F. Lemmer, editors, *Uncertainty in Artificial Intelligence.* North-Holland, 1986.

[13] J. P. Kassirer and G. A. Gorry. Clinical problem solving: A behavioral analysis. *Annals of Internal Medicine*, 89:245–255, 1978.

[14] Doug Lenat, Mayank Prakash, and Mary Shepherd. CYC: Using common sense knowledge to overcome brittleness and knowledge acquisition bottlenecks. *AI Magazine*, 6(4):65–85, 1986.

[15] Robert K. Lindsay, B. G. Buchanan, E. A. Feigenbaum, and J. Lederberg. *Applications of Artificial Intelligence for Organic Chemistry: The DENDRAL Project.* McGraw-Hill, 1980.

[16] William J. Long. Reasoning about state from causation and time in a medical domain. In *Proceedings of the National Conference on Artificial Intelligence*, pages 251–254, 1983.

[17] W. A. Martin. Interactive systems—theories of implementation. Course notes for Knowledge-Based Application Systems, 1977.

[18] R. A. Miller, M. A. McNeil, S. M. Challinor, F. E. Masari, Jr., and J. D. Myers. The Internist-1/Quick Medical Reference project—status report. *Western Journal of Medicine*, 145:816–822, 1986.

[19] Randolph A. Miller, Harry E. Pople, Jr., and Jack D. Myers. Internist-1, an experimental computer-based diagnostic consultant for general internal medicine. *New England Journal of Medicine*, 307:468–476, 1982.

[20] Marvin Minsky. Why people think computers can't. *AI Magazine*, pages 3–15, 1982.

[21] Ramesh S. Patil. Causal representation of patient illness for electrolyte and acid-base diagnosis. TR 267, Massachusetts Institute of Technology, Laboratory for Computer Science, 545 Technology Square, Cambridge, MA, 02139, October 1981.

[22] Stephen G. Pauker, G. Anthony Gorry, Jerome P. Kassirer, and William B. Schwartz. Towards the simulation of clinical cognition: Taking a present illness by computer. *American Journal of Medicine*, 60:981–996, 1976.

[23] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, 1988.

[24] Edward H. Shortliffe and Bruce G. Buchanan. A model of inexact reasoning in medicine. *Mathematical Biosciences*, 23:351–379, 1975.

[25] Reid Gordon Simmons. *Generate, Test, and Debug: A Paradigm for Solving Interpretation and Planning Problems.* PhD thesis, Massachusetts Institute of Technology, 1988.

[26] William Swartout and Robert Balzer. On the inevitable intertwining of specification and implementation. *Communications of the ACM*, 25(7):438–440, 1982.

[27] William R. Swartout. XPLAIN: A system for creating and explaining expert consulting programs. *Artificial Intelligence*, 21:285–325, 1983.

[28] P. Szolovits, J. P. Kassirer, W. J. Long, A. J. Moskowitz, S. G. Pauker, R. S. Patil, and M. P. Wellman. An artificial intelligence approach to clinical decision making. TM 310, Massachusetts Institute of Technology, Laboratory for Computer Science, 545 Technology Square, Cambridge, MA, 02139, September 1986.

[29] Peter Szolovits and William J. Long. The development of clinical expertise in the computer. In Peter Szolovits, editor, *Artificial Intelligence in Medicine*, volume 51 of *AAAS Selected Symposium Series*, pages 79–117. Westview Press, Boulder, Colorado, 1982.

[30] Randy L. Teach and Edward H. Shortliffe. An analysis of physician attitudes regarding computer-based clinical consultation systems. *Computers and Biomedical Research*, 14:542–548, 1981.

[31] L. L. Weed. *Medical Records, Medical Education and Patient Care.* Case University Press, Cleveland, Ohio, 1969.

[32] Michael P. Wellman. Formulation of tradeoffs in planning under uncertainty. TR 427, Massachusetts Institute of Technology, Laboratory for Computer Science, 545 Technology Square, Cambridge, MA, 02139, August 1988. Revised version of a Ph.D. dissertation, July 1988.

[33] V. L. Yu, B. G. Buchanan, E. H. Shortliffe, S. M. Wraith, R. Davis, A. C. Scott, and S. N. Cohen. An evaluation of the performance of a computer-based consultant. *Computer Programs in Biomedicine*, 9:95–102, 1979.