Reconstructive Explanation: Explanation as Complex Problem Solving

Michael R. Wick*
Computer Science Department
University of Minnesota
Minneapolis, MN 55455

William B. Thompson
Computer Science Department
University of Minnesota
Minneapolis, MN 55455

Abstract

Existing explanation facilities are typically far more appropriate for knowledge engineers engaged in system maintenance than for end-users of the system. This is because the explanation is little more than a trace of the detailed problem-solving steps. An alternative approach recognizes that an effective explanation often needs to substantially reorganize the line of reasoning and bring to bear additional information to support the result. Explanation itself becomes a complex problem-solving process that depends not only on the line of reasoning, but also on additional knowledge of the domain. We present a new computational model of explanation and argue that it results in significant improvements over traditional approaches.

1 Introduction

A computer is generally poor at explaining its problem solving to a human user. Early work on expert systems suggested an explicit knowledge base of expert-defined, problem-solving rules might be used to help such a program explain its reasoning. During the past decade, research has been conducted on ways of using this knowledge base to explain the expert system's actions and conclusions. Although significant advances have been made, explanations still suffer from several important flaws. The underlying premise of previous work is that the basis of the explanation is the trace of the expert system's *line of reasoning.* We believe another approach is possible that, for certain audiences, will overcome many of the problems evident in earlier explanations. A human expert, when asked to account for complex reasoning, rarely does so exclusively in terms of the actual process used to solve the problem [Ericsson and Simon, 1984]. Instead, an expert tends to reconstruct a "story" that accounts for the problem solving. This story reflects the expert's *line of explanation* [Paris *et al,* 1988] that is not necessarily the same as the original line of reasoning. For example, consider the following line of reasoning taken by an inspector attempting to find the cause of the excessive load on a concrete dam (based on [Franck, 1987]).

> *...the debris on top of the dam suggests a recent flood. The water markings on the abutments do too. I suspect the flood is the cause of the excessive load. No, the duration of the flood wasn't long enough. Sometimes settlement has these same features. Perhaps settlement is involved. That would account for the high uplift pressures suggested by the slow drainage over time. But the damage to the drainage pipes isn't right. It must be erosion causing the dam to drop more at the toe. Yes, erosion is causing the excessive load...*

Note that the inspector is using a heuristic, data-driven problem-solving process. Later, the field inspector is asked to explain the reasoning that led to the conclusion.

> *...the symptoms led me to believe the problem is internal erosion of soil from under the dam. See, erosion would cause the selectively broken pipes under the dam, therefore slowing drainage and causing high uplift pressures that cause the dam to slide downstream...*

Notice how the line of explanation is different from the line of reasoning. During problem solving, the line of reasoning is directed to *settlement* through a heuristic association with *flood,* and then on to *erosion.* However, the line of explanation moves to *erosion* directly. The line of explanation is not simply a version of the line of reasoning pruned for dead-ends. The heuristic association between *flood* and *settlement* that eventually led to the conclusion *erosion* has been replaced by relationships that bond the symptoms directly to *erosion.*

Data introduction is another interesting feature of this explanation. During explanation, evidence not used during problem solving is introduced as additional support. This includes not only the underlying causality of many of the items, but also the introduction of new symptoms that further support the final conclusion (i.e. the movement of the dam). This type of data introduction is common in domains marked by *nonexhaustive problem solving.* In such domains, an expert will use a small set

*Present Address: Computer Science Department, Washington State University, Pullman, Washington 99164-1210.

This research is sponsored in part by the National Science Foundation, grant number IRI-8715623.

of cues from the data to reach a conclusion. Once this conclusion has been made, the expert will support it with additional data items. In some explanations, the initial data cues are replaced with new more directly supporting data. In our example, the triggering data (i.e. the duration of the flood) is dropped as it is not needed to directly support the conclusion.

As illustrated by this example, the line of explanation and the line of reasoning are often considerably different in both form and content. In contrast to previous work, our research aims at creating the explanation as a product of a problem-solving activity largely distinct from the expert system's problem-solving process. This breaks the tight bond between explanation and problem solving. With this bond broken, the explanation system has freedom to reconstruct the explanation to create a more direct account of the expert system's conclusion.

2 Previous Work

Mycin [Shortliffe, 1976] was one of the first systems to explain its actions. Mycin provided two basic explanation queries: *why* and *how.* These two queries form the foundation of nearly all explanation facilities to date.

Swartout [Swartout, 1983] introduced a system, called XPLAIN, explicitly designed to attack the problem of explanation. Swartout used *domain principles* and *domain rationales* to record the designer's rule justifications by using an automatic programmer to build the expert system. The XPLAIN system produced excellent explanations. However, the information presented appears to be best suited for a knowledge engineer.

Clancey [Hasling *et* a/., 1984] has built an explanation system that augments the facility provided by Mycin. Clancey's system, NEOMYCIN, shifts the focus from the domain knowledge to the strategic problem-solving knowledge, NEOMYCIN is capable of generating *why* and *how* explanations about the strategy used to solve the problem.

Many previous systems, for example XPLAIN, do not present a strict one-to-one mapping of the problem-solving steps but allow some of the steps to be omitted based on the type of user (i.e. knowledge engineer or domain expert). Other systems, such as that by Wallis and Shortliffe [Wallis and Shortliffe, 1982], include measures of "complexity" and "importance" for pruning steps from the trace. However, none of these systems includes the ability to completely reconstruct the explanation or to introduce new data not originally used by the expert system.

Tanner [Tanner and Josephson, 1988] has developed on a system to construct justifications that are designed, not necessarily to follow the problem-solving process, but instead to explicitly connect with the user's understanding of the problem. Paris [Paris, 1987] has found that explanations not only vary in abstraction level according to the user, but they also vary in content and emphasis. This work strongly suggests that an expert system must be able to change the content of the explanation, if it is to be practical for varying users. Moore [Moore and Swartout, 1988] is also working on changing the content of an explanation based on the user. Her work centers on the use of rhetorical techniques to plan an explanation of the execution trace to produce a context for follow-up questions.

Our research differs from related work in several ways. Most significantly, our research aims at using knowledge other than the execution trace as the basis for the explanation. Previous work concentrated on methods for translating, presenting, or augmenting the execution trace. Our work focuses on replacing the trace with an equally valid line of explanation.

3 Different Kinds of Explanation

The nature of an effective explanation depends heavily on the user. Knowledge engineers and others involved in the design and maintenance of an expert system require an explanation facility that elaborates on precisely what the system did to accomplish a specific result. This reflects the common thought that "justification must remain true to the expert system". However, an explanation for an end-user is intended to increase the user's confidence in the system and to aid the user in understanding the consequences of the system's conclusion. A system designer clearly needs a traced-based explanation that accurately reflects the line of reasoning used in the expert system. This line of reasoning may be inappropriate, however, for an end-user. A line of reasoning often proceeds to a conclusion via an obscure and indirect path. For instance, as illustrated in section 1, many complex domains involve heuristic reasoning [Clancey, 1985]. An effective explanation of the conclusion *erosion,* although arrived at from a heuristic association with *flood,* may not only require a substantial reorganization of the line of reasoning, but may require the use of additional supporting information not part of the original reasoning process. Generating such an explanation is possible only within a *reconstructive explanation*[1] paradigm in which the explanation is reconstructed by an active, problem-solving process.

There are obvious costs associated with the adoption of a reconstructive explanation strategy. However, these costs may not be as great as might at first be supposed. A clearer separation between problem solving and explanation reduces the need to trade problem-solving competence for comprehensibility that often arises with conventional explanation systems. Another possible problem with reconstructive explanation is the potential inconsistency between problem solving and explanation. However, better separation of problem solving and explanation might improve consistency. A recent study has shown that non-experts are not likely to catch reasoning errors when presented with trace-based explanations [Erdman, 1983]. This may be because a non-expert, such as an end-user, often will not catch reasoning errors in a difficult to understand line of reasoning. Reconstructive explanation can aid the end-user in a better understanding of the problem and thus provide a basis for the user

[1]It should be noted that this use of "reconstructive explanation" differs from an earlier use [Paris *et* a/., 1988]. The current use is more restrictive than the former, corresponding to what was previously called "plausible explanation."

independently evaluating a system's actions. Overall, quality end-user explanation is not free and the designers of an expert system must determine if they require extensive end-user explanations.

4 Intuitive Support for Reconstructive Explanation

As reconstructive explanation is significantly different from previous explanation paradigms, we present some intuitive support for its use. Research in psychology has discovered that, in certain situations, recall is reconstructive [Dawes, 1964]. That is, during recall of an event, details are filled in that are not available from the memory of that particular event. This same notion of reconstruction has be shown to carry over to complex problem solving [Ericsson and Simon, 1984]. In expert problem solving, many of the details of how and why things happened are not available from a memory of the problem solving [Anderson, 1982]. When asked to report this information, the expert will reconstruct an explanation that integrates the elements of a partial memory trace with the memory of other related entities (for example, textbook information). We believe this freedom to reconstruct an explanation based on information in addition to the information and processes used during problem solving is in part responsible for the high quality of human explanations. Reconstructive expert system explanation is the study of how to give an expert system this reconstructive ability.

5 The Rex System

REX (Reconstructive Explainer) is a test-bed system capable of producing reconstructive explanations for expert systems. REX is designed to answer retrospective queries in an attempt to obtain the end-user's ratification of the expert system's conclusion. Such retrospective queries have been shown to be a useful context for reconstructive explanation techniques [Ericsson and Simon, 1984]. REX provides an explanation of the movement within the competitor set of conclusions, thus showing how the final conclusion was reached.

5.1 A Model of Reconstructive Explanation

The REX system is built on a model of reconstructive explanation that maps the execution of the expert system onto a textbook representation of the domain. Here, a textbook representation is simply a representation of the information presented in human explanations, much of which comes from domain textbooks. The process of explanation then becomes the process of mapping over key elements from the execution trace and expanding on them using the more structured textbook knowledge. This should not be confused with "decompiling" the rules used by the expert system. Decompiling results in an explanation of the relationship between the antecedent and consequent of a rule. Reconstruction results in textbook knowledge that accounts for the data uncovered by the expert system.

The execution trace is passed from the expert system to REX where it is mapped into a high-level specification of expertise that represents the knowledge required for this problem-solving task. The specification points to structures in the textbook knowledge base that represent methods and relationships involved in performing each particular problem-solving task. A search is conducted within the textbook knowledge base to find the information necessary to answer the end-user's query.

In the traditional approach to explanation the execution trace is mapped directly to the explanation text. Sometimes pruning is done to remove unnecessary information and auxiliary knowledge (called support knowledge) is added. However, the line of the explanation is based exclusively on the trace or line of reasoning. In our reconstructive explanation approach the execution trace is mapped through a high-level specification to the textbook knowledge of the domain. As the trace is mapped through the specification, information on the processes and reasons used by the expert system (the "how" knowledge) is filtered out. This leaves only information on the data used during problem solving (the "what" knowledge). Using this "what" knowledge as an index, textbook methods and relations are found that represent well-organized "how" knowledge. This textbook information, as opposed to the more obscure information found in the trace, forms the content and organizational structure of the explanation text.

An obvious question arises. Why go through all the extra work of mapping the trace into the textbook knowledge? Why not simply use the "textbook justifications" attached to the rules in the trace? The answer rests in the realization that explanation is a complex problem-solving task in its own right, requiring at a minimum a significant reorganization of the knowledge used for problem solving. For example, in medical school, knowledge is presented and explained as *symptoms given disease.* In the real world, problems are solved as *disease given symptoms.* However, conclusions are still explained as *symptoms given disease* as the explanation is then easier to understand and follow. Likewise, when an expert is asked to explain complex problem-solving methods, what can be given is the textbook method of solving the problem, instantiated for the specific case at hand. This method explains the process for the same reason that it was used to teach the process, it is easier to understand than the more obscure and ad hoc method used during real problem solving.

This raises yet another question. If the textbook methods are so easy to explain, why not use them directly to solve the problem? The answer rests in the realization that real-world problem solving is also a complex task. Experts often proceed from symptoms to conclusions through long diverted paths. It is rarely possible to identify what method of solving the problem is appropriate from the beginning. It is much easier to reconstruct an appropriate method once the answer is known. Thus, explanation and problem solving are two largely distinct tasks, each requiring its own methods and knowledge.

5.2 Specification of Expertise

Before describing how REX reconstructs an explanation, it is first necessary to briefly describe the representation used for the high-level specification. The specification represents what knowledge is required to solve the problem. In particular, this representation of expertise takes the form of a graph of hypotheses, each connect to other hypotheses by one or more "transitions" [Johnson *et ai,* 1987]. Associated with each transition are two sets. A set (called the *condition* set) of cues defines the data that must be found for the transition to be valid. In other words, this set represents information from the problem solving that can lead to a movement between the two hypotheses. Secondly, another set (called the *goal* set) defines the goals that must be posted in moving between the two hypotheses. These two sets combine to define what knowledge (cues and goals) is required to move between two hypotheses in the knowledge specification.

At first, this representation may sound like a representation of how to solve the problem. However, the representation is neither procedural or deterministic. In other words, each transition tells what information is required to move between two hypotheses, but does not enforce how that information will be used. For example, the *condition* and *goal* elements of a transition are supersets of the cues and goals needed to make the transition. Different subset(s) of cues and goals can be used to move between the two hypotheses. The following section will illustrate the use of such a specification for reconstructing explanations.

5.3 Reconstruction of How from What

To illustrate the process of explanation, consider the line of reasoning given in section 1. In that example, the domain expert was trying to identify the cause of the excessive load on a concrete darn. In the explanation, the expert was attempting to answer how that cause was determined. The expert's problem-solving leaves a trace of data corresponding to symptoms and inferences. The trace for this example contains: *debris on dam, water marks, drainage, uplift pressure,* and *broken pipes.* In REX, the data in this trace are used to "activate" the same data in the high-level specification of the knowledge required to solve the problem. This enables REX to determine what data cues were used by the expert system in moving to the conclusion. The line of reasoning illustrated in section 1 follows a path from the initial empty hypothesis through the hypothesis *flood,* onto the hypothesis *settlement* and stopping at the final solution *erosion.* However, the line of explanation (as shown in section 1) moves from the initial empty hypothesis directly to *erosion,* using the data cues *uplift pressure, drainage, broken pipes* and *sliding.* REX, when given access to the expert system to find additional supporting knowledge that was not activated from the expert system's problem-solving trace, reconstructs an explanation that closely resembles this line of explanation. This reconstruction involves three core elements of the REX design: the textbook knowledge base, the explainer, and the story teller. The following paragraphs will describe how each of these elements helps reconstruct a line of explanation for the concrete dam example.

```
CUE uplift
   Value          true
   Type           direct
   Name           the high uplift pressures acting on
                  the dam
   Nickname       uplift pressures
   Valuename

HYPOTHESIS erosion
   Value          true
   Name           the erosion of soil from under the dam
   Nickname       erosion
   Valuename      * hypothesis*

GOAL det-cause
   Name           determine causal relationships
   Nickname       determine causes

CUE SCRIPT erosion-to-sliding
   Uses         : (<drainage> <uplift> <sliding>)
   Supports        <erosion>
   Achieves       det-cause
   Vconstraint    (and <drainage> <uplift>
                  <sliding>)
   Text           (<erosion> causes <broken-pipes>
                  causing <drainage> resulting in
                  <uplift> and in turn <sliding>)
GOAL SCRIPT causal
   Holds           (<det-cause>)
   Text         :  (simply <det-cause>)
```

Figure 1: Sample textbook knowledge representations.

The Textbook Knowledge Base is represented in REX as a collection of relationships between cues, hypotheses, and goals as illustrated in Figure 1. The cues, hypotheses, and goals themselves represent the domain objects that other relationships and methods manipulate. In REX, each relationship is represented as a cue script and each method is represented as a goal script as shown in Figure 1. Each frame and script has text slots for English presentation.

Using the representations illustrated in Figure 1, a transition from one hypothesis to another is possible when a method and relationships are found such that each goal and cue used is a member of the *goal* and *condition* sets defined between the two hypotheses in the knowledge specification graph. In REX, the structure built by combining the method and the relationships is called an *explanation structure* as it serves as an explanation of the movement between the hypotheses.

The Explainer is responsible for constructing the line of explanation that will eventually be presented to the end-user. This line of explanation represents a movement from the initial problem-solving state to the final conclusion reached by the expert system. In REX, this corresponds to finding a path through the knowledge specification from the conclusion of the expert system to the empty hypothesis. Each transition in this path must be supported by the existence of a valid explanation

```
LOE        ::= {problem description} {problem statement}
               Story Resolution
Story      ::= Setting Theme Plot Resolution |
               Setting Theme Plot Resolution Story
Setting    ::= {hypothesis}
Theme      ::= Event Goal
Event      ::= {cue} | {cue} Event
Goal       ::= {make initial hyp} | {refute hyp} |
               {generalize hyp} | {support hyp} |
               {refine hyp}
Plot       ::= Strategy Relations Outcome
Strategy   ::= {goal} | {goal} Strategy
Relations  ::= {textbook relation} | {textbook relation}
               Relations
Outcome    ::= {hypothesis}
Resolution::= {hypothesis}
```

Figure 2: The grammer used by REX.

structure. REX uses the A* algorithm to search through a space of knowledge specification transitions for which a valid explanation structure has been found. The search is carried out backwards from the final conclusion of the expert system towards the empty hypothesis. Each state in this search corresponds to an emerging line of explanation that uses certain cues and a hypothesis as data, establishes other cues and a hypothesis as conclusions and traverses certain edges in the specification. Transitions between states in the A* search correspond to expanding the bottom hypothesis by finding each edge with a valid explanation structure that moves to this hypothesis. As the precise explanation structure chosen will determine the cues included in the explanation, a separate transition in the A* search is constructed for each valid explanation structure on each incoming edge of the bottom hypothesis. A complete line of explanation represents a path from the initial problem state (the empty hypothesis) to the final conclusion reached by the expert system.

The Story Teller takes the path found by the explainer and formats it for English presentation using the grammer of Figure 2, thus creating what is called a *story tree.* Figure 3 shows the story tree for our example. The REX system output of this story tree is as follows:[2]

> We have a concrete dam under an excessive load. I attempted to find the cause of the excessive load. Not knowing the solution and based on the broken pipes in the foundation of the dam, and the downstream sliding of the dam, and the high uplift pressures acting on the dam, and the slow drainage of water from the upstream side of the dam to the downstream side I was able to make an initial hypothesis. To achieve this 1 used the strategy of striving to simply determine causal relationships. In attempting to determine causes, I found that the internal erosion of soil from under the dam causes broken pipes causing slow drainage resulting in uplift and in turn sliding. This led me to hypothesize that internal erosion was the cause of the excessive load. Feeling confident in this solution, I concluded that the internal erosion of soil from under the dam was the cause of the excessive load.

[2]The verbose nature of the English output is a result of our focus on the content and structure of the story tree and not on its presentation.

The expert system, using a reconstructive explanation system, is able to present a line of explanation that leads directly to the solution. Whereas the expert system using a traditional explanation system would be restricted to a line of explanation moving first to *flood,* through *settlement* to *erosion.* Even when the explanation system is not given access to the expert system and thus can not ask for additional supporting data (such as *sliding)* the reconstructive paradigm can still create a more direct explanation than is possible within the traditional paradigm. REX can find the shortest path of "activated" data from the solution hypothesis to the initial empty condition. In other words, REX can find the most direct path to the solution using only information uncovered by the expert system during problem solving. In our example, this path is found by moving from the initial empty set directly to *settlement* and on to *erosion.* This line of explanation, although less direct than the previous line of explanation, is still more direct than the path followed by the traditional explanation paradigm as it by-passes the need for the heuristic association with *flood.*

6 Significance

Reconstructive explanation is a significantly new approach to the automatic generation of expert system explanations. The feasibility of using the reconstructive explanation paradigm has been shown by the REX system.

Such a reconstructive explanation paradigm has several advantages that show its desirability: (1) The textbook methods and relations used to integrate the information uncovered during problem solving serve to reorganize the flow of the explanation to be more direct. (2) Different methods and relations can be used to allow the explanations to be tailored to the needs of specific user types. (3) A reconstructive explanation system can provide independent feedback on the performance of the expert system. For example, if the explanation system can not find an explanation for the conclusion, this could suggest an error in the problem solving of the expert system. (4) An expert system with a reconstructive explanation facility will have the ability to use one approach for problem solving and another for explanation. Thus, the system can be implemented with less concern for the tradeoff between problem-solving ability and end-user explanation. (5) Under certain constraints, a reconstructive explanation has the freedom to present multiple lines of explanation leading to the same conclusion. (6) Reconstructive explanation provides more flexibility than conventional explanation systems allowing explanations to be built on information other than a subset of the execution trace.

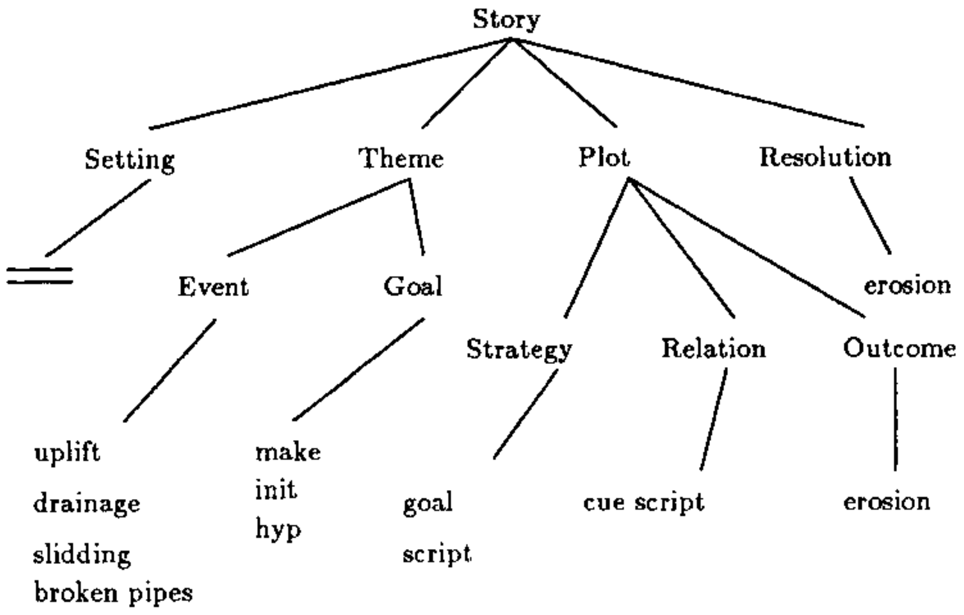

Figure 3: An example reconstructed explanation story.

References

[Anderson, 1982] J.R. Anderson. Acquisition of cognitive skill. *Psychological Review,* 89(4):369-406, 1982.

[Clancey, 1985] W.J. Clancey. Heuristic classification. *Artificial Intelligence,* 27:289-350, 1985.

[Dawes, 1964] R. Dawes. Cognitive distortion. *Psychological Reports,* 14:443-459, 1964.

[Erdman, 1983] P.H. Erdman. *A Comparison of Computer Consultation Programs For Primary Care Physicians: Impact of Decision Making Model and Explanation.* PhD thesis, University of Wisconsin - Madison, 1983.

[Ericsson and Simon, 1984] K. A. Ericsson and HA. Simon. *Protocol Analysis: Verbal Report as Data.* MIT Press, Cambridge, Massachusetts, 1984.

[Franck, 1987] B. Franck. *Preliminary Safety and Risk Assessment For Existing Hydraulic Structures -An Expert Systems Approach.* PhD thesis, Mechanical Engineering Department, The University of Minnesota, Minneapolis, 1987.

[Hasling *et ai,* 1984] D.W. Hasling, W.J. Clancey, and G. R.ennels. • Strategic explanation for a diagnostic consultation system. *Int. J. Man-Machine Studies,* 20:3 19, 1984.

[Johnson *et ai,* 1987] P. Johnson, I. Zualkernan, and S. Garber. Specification of expertise. *Int. J. Man-Machine Studies,* 26:161-181, 1987.

[Moore and Swartout, 1988] J.D. M oore and W.R. Swartout. A reactive approach to explanation. In *Proceedings of the AAAI Workshop on Explanation,* pages 91-94, August 1988.

[Paris, 1987] C.L. Paris. Combining discourse strategies to generate descriptions to users along a naive/expert spectrum. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence,* pages 626-632, 1987.

[Paris *et ai,* 1988] C.L. Paris, M.R. Wick, and W.B. Thompson. The line of reasoning versus the line of explanation. In *Proceedings of the AAAI Workshop on Explanation,* pages 4-7, 1988.

[Shortliffe, 1976] E.H. Shortliffe. *Computer-Based Medical Consultations: MYCIN.* New York: Elsevier/North Holland, 1976.

[Swartout, 1983] W.R, Swartout. XPLAIN: a system for creating and explaining expert consulting programs. *Artificial Intelligence,* 21:285-325, September 1983.

[Tanner and Josephson, 1988] M.C. Tanner and J.R, Josephson. Justifying diagnostic conclusions. In *Proceedings of the AAAI Workshop on Explanation,* pages 76-79, August 1988.

[Wallisand Shortliffe, 1982] J. Wallis and E. Shortliffe. Explanatory power for medical expert systems: studies in the representation of causal relationships for clinical consultations. *Methods of Information in Medicine,* 21:127-136, 1982.