Proceedings of the Fourth Florida Artificial Intelligence Research Symposium, pp. 226–231, Cocoa Beach, FL, April 1991.

Learning Indices for Schema Selection*

Sambasiva Bhatta and Ashwin Ram
College of Computing
Georgia Institute of Technology
Atlanta, Georgia 30332-0280

Abstract

In addition to learning new knowledge, a system must be able to learn when the knowledge is likely to be applicable. An index is a piece of information which, when identified in a given situation, triggers the relevant piece of knowledge (or schema) in the system's memory. We discuss the issue of how indices may be learned automatically in the context of a story understanding task, and present a program that can learn new indices for existing explanatory schemas. We discuss two methods using which the system can identify the relevant schema even if the input does not directly match an existing index, and learn a new index to allow it to retrieve this schema more efficiently in the future.

1 Introduction: The index learning problem

Knowledge plays an important role in AI systems, both systems that understand natural language stories and those that solve problems. However, we cannot expect systems to have all the necessary knowledge *a priori*. We need systems that can learn through experience and use the acquired knowledge in subsequent situations. Furthermore, since the amount of knowledge required for and acquired during such tasks is very large, a system needs the right pointers (otherwise known as *indices*) to different pieces of knowledge so that they can be retrieved efficiently at the right time. An index is a piece of information which, when identified in a given situation, triggers the relevant piece of knowledge in the system's memory. In addition to learning new knowledge, a system needs to learn indices for it so that it can retrieve and use the knowledge efficiently and effectively.

Indexing is hard because there is no deterministic way of deciding what should be an index to a given piece of knowledge. We cannot anticipate in advance all the situations in which a given piece of knowledge will be required. Furthermore, knowledge is not static. An index that does well at one point of time may not be any good at a different time when knowledge grows (or gets modified). In addition to characterizing what makes a good index, we also need to develop mechanisms by which old indices are refined or new indices are learned. This paper addresses a fundamental and important issue in learning: *What parts of knowledge learned should be used as indices to that knowledge? How can a system automatically learn these indices?*

It is difficult to enumerate all the possible situations that could arise and to form the required indices for each of these situations. Our approach is to choose indices for the available knowledge depending on the purpose for which they are intended (i.e., satisfying the constraints of the tasks at hand). This is augmented with a mechanism for learning new indices, which improves the system's ability to retrieve the relevant knowledge in different situations. This allows the system to compensate for gaps in the system's initial knowledge base, as well as adapt to changing circumstances.

*This research was partly supported by the National Science Foundation under contract IRI-9009710.

2 Example

Consider a schema-based story understanding system that uses causal schemas, such as *explanation patterns (XPs)* (Schank 1986; Ram 1990a), to understand stories, and that can learn a new schema based on a novel story it has processed (e.g., GENESIS (DeJong and Mooney 1986), AQUA (Ram 1989; Ram 1990b)). What should the system use as an index to the new schema or XP? A simple approach would be to record the information in the first sentence of the story as an index to the new XP. But in general that need not necessarily be a *good* index. For example, consider the following story:

S_1: Jack was driving home.
S_2: He was traveling above the speed limit.
S_3: He hit a parked car.
S_4: He was badly injured and taken to hospital.

Suppose that the system has an explanation pattern, XP-VEHICLE-ACCIDENT, describing the causality of a typical vehicle accident scenario. If DRIVE, the action underlying the first event S_1, was not already recorded as an index to XP-VEHICLE-ACCIDENT, the system could learn it as an index to the XP. But DRIVE is not a good index to XP-VEHICLE-ACCIDENT, even though it is one of the preconditions of the XP, because this will cause the XP to be triggered whenever "driving" is involved in an event. This would complicate the *schema selection problem*. The problem is that DRIVE, although a precondition for XP-VEHICLE-ACCIDENT, is not necessarily *predictive* of an accident. If the index were instead "driving fast" or "driving rashly," it is more likely that an accident might follow, and hence more likely that XP-VEHICLE-ACCIDENT would apply to the situation. In other words, "driving rashly" would form a good *predictive index* to XP-VEHICLE-ACCIDENT.

We know this because we have general knowledge about when an accident is likely to occur that acts as an *explanation* or a *justification* in selecting an index to our vehicle accident schema. In forming a new index, the system should be able to explain, or causally link, the index to the other parts of the XP, or to some general knowledge that the system already has. The intuition here is that the same kind of knowledge used in understanding, namely, causal schemas or XPs, is also useful in deriving an appropriate new index. A learning method based on *explanations* would come up with better and more appropriate indices than *ad hoc* criteria.

In this paper, we describe how new indices can be learned and used to retrieve appropriate schemas, such as XPs, for understanding stories. Explanation-based techniques for learning have been proposed by several people (e.g., Mitchell, Keller, and Kedar-Cabelli (1986); DeJong and Mooney (1986)) in several domains. Barletta and Mark (1988) argue that the explanations or justifications are essential and useful for learning of good indices for case-based problem solving. We extend some of their results to the task of story understanding.

Furthermore, learning in most systems is treated as a separate phase in the reasoning process, usually invoked after the main task of the system is completed. However, since an index is determined by the purpose for which it is intended, we hypothesize that the learning of an index would be most effective if integrated with the task or subtask where the index will be used.

3 Methods for schema selection and learning

The schema selection problem is part of the general problem of indexing faced by all schema-based and case-based reasoning systems. Early story understanding systems that use schemas include SAM (Cullingford 1978), FRUMP (DeJong 1979) and IPP (Lebowitz 1980). All these systems assume that the first or an early sentence in a story is the main source of indices for retrieving relevant schema or other knowledge structures. But it is not uncommon for stories to start with a description of several different preconditions, eventually leading to a main event for which there is a schema available that explains that event. However, the schema may not have any indices that are present in the leading sentences (or events). These preconditions may not be available as indices because the system's knowledge is incomplete. In the worst case, the schema or XP may not be triggered at all, even later in the story, without considerable inference. This is precisely the situation where we want the system to learn a new index to the associated schema if one exists, or else learn both a schema and an index to it.

For example, consider the driving story in section 2. Since the first event, S_1, is too general, it would not be used as a predictive index to XP-VEHICLE-ACCIDENT. If a later event ("driving fast" or "hitting a car") is known to be an index for XP-VEHICLE-ACCIDENT, the system could understand the story by *deferring* the schema selection process until later in the story, and then going back and integrating the earlier sentences with the retrieved XP. This method is called *deferred schema selection.*

To take the example further, unless "driving fast" or "hitting a car" are already known to be indices to XP-VEHICLE-ACCIDENT, it would be difficult for a conventional schema-based understanding program to retrieve this XP. Even S_2 does not imply that there was an accident; rather, this follows from the fact that "Jack was driving a car when he hit a parked car," which is not explicitly stated in the story. In such cases where indices are not explicitly mentioned in the story but need to be inferred, we want the system to be able to *infer* an index to XP-VEHICLE-ACCIDENT, and hence retrieve the XP rather than fail. This method is called *inferred schema selection.* We also want the system to learn a new index (here, "driving rashly") as a result of having made these inferences, and thus to get better at recognizing when to retrieve the XP.

We propose two new methods that integrate schema selection, a fundamental subtask in all schema-based processing systems, with index learning as well as with schema learning. In this paper our emphasis will be on index learning.

3.1 Deferred schema selection and learning

Given a story input, if the first sentence does not trigger any schema that can predict and explain the story, the system continues to accumulate input sentences from the story with a hope of encountering a sentence later in the story that will trigger a schema, deferring the decisions to either fail from the task of understanding the sentences or to use expensive inference mechanisms to understand them. When a schema is eventually selected, the system learns a new index to the schema that will allow it to retrieve the schema earlier when a similar story is read in the future. A simple heuristic is to generalize the first sentence to locate an index. We discuss the simple method first, and a better method later in the paper.

Consider the previous example. Following the deferred schema selection algorithm, if the first sentence S_1 does not trigger XP-VEHICLE-ACCIDENT, the system would simply continue to process the later sentences, delaying the task of understanding previous sentence(s) for later. S_2 does not trigger the XP, either. But S_3 contains an index to the XP-VEHICLE-ACCIDENT ("hitting a car" can be a situational index (Ram 1989) to this XP), and it triggers this schema. XP-VEHICLE-ACCIDENT suggests that the precondition for an accident is "driving" by which the first deferred sentence (S_1 in this story) is connected. S_2 further qualifies "driving," the action in S_1. The XP also predicts that the person could be injured and needs to be taken to a hospital which will allow the system to understand the rest of the story (sentence S_4) without much inference using the usual top-down understanding mechanism used in script- or schema-based understanding programs such as SAM or AQUA.

Using the simple heuristic mentioned above, the system generalizes the first of the deferred sentences (S_1), and installs the DRIVE action as an index to the XP-VEHICLE-ACCIDENT. Although this heuristic does not always lead to good indices (e.g., as in this case), it is efficient and easy to use. A better method is discussed later.

3.2 Inferred schema selection and learning

Given a story input, if the first few sentences do not directly trigger any XPs, the system draws inferences from these sentences that might trigger a relevant XP via an existing index, and learns a new index from the inferences that lead to that index. Two important issues that spring from this method are: How far should the system continue the inference chain if no XP has been triggered? And, what knowledge should be used to constrain the inference?

For example, suppose the first sentence in the above story were as follows:

S_1': Jack was driving rashly.

Even without a "driving rashly" index, the system can infer that Jack might get into an accident, and thus be able to retrieve the XP-VEHICLE-ACCIDENT. This inference requires general knowledge about the words in the story (similar to (Charniak 1978)), and about the concepts in this domain (such as DRIVE typically involves a driver, a vehicle as an instrument, etc.) Here, the system might infer that whenever someone is driving rashly, there is a strong possibility that an accident might occur. The system infers (predicts) certain state(s) or event(s) to be likely to follow in the story, and one of those predicted events ("accident") triggers an XP (XP-VEHICLE-ACCIDENT) that then helps in understanding the story.

In order to decide when to perform this inference, a simple heuristic, corresponding to the simple heuristic for deferred schema selection, would be to do the inference only if the first sentence does not directly trigger any schema. The inference can be as simple as "chaining" (as in (Rieger 1975)). This process uses causal rules to construct inferences by constructing chains of inferences based on these rules. Although this method helps the system retrieve XPs and thus come to a better understanding of the story, this improvement comes at the expense of possible combinatorial explosion of inferences. One way to speed up the inference process is to use causal rules that are not so primitive (as in Rieger's program), but rather fairly abstract rules that connect events at the level of actions. Such rules can be viewed as simple but very general explanations. Alternatively, some other constrained inference process may be used, in particular, knitting (bidirectional chaining). This can be guided by the indices available, because ultimately an inferred fact has to trigger a schema or an XP by matching an index.

4 The integrated approach

Considering the two mechanisms, it is clear that a combination of the two methods is best. Furthermore, rather than using these methods as two independent mechanisms for indexing and retrieval, it may be necessary to integrate them. For example, consider the situation in which the inferences required by the inferred schema selection method to derive an index for a schema involves more than one sentence from the story. In this case, the deferred schema selection method is also necessary. For example, in the driving story discussed in section 2, the first two sentences are required in order to predict the "driving fast" index. This requires the first sentence to be deferred, and then the inferences to be drawn from both sentences. The combined method is shown in figure 1. The method relies on the intuition that if a schema is triggered late in a story, it is likely that the preceding (deferred) sentences of the story could have been used as indices to the XP that eventually turns out to be applicable had the system known these indices in advance. Thus the events mentioned in these sentences should be used to create new indices. However, we can not arbitrarily install these events as indices to the XP unless we have an explanation of the connections between the events and the XP (more precisely, between the events and one of the available indices to the XP, which are known to constitute predictive triggering conditions for the XP.) At this point, the integrated method makes use of the explanations (in the form of causal rules) in order to connect the deferred sentences and one of the available indices for the XP (steps 3–5 in figure 1). By "connect" we mean to establish links between the roles (actor, theme etc.,), objects, and concepts of one sentence to those of another in order to realize "coherence" among the events described in those sentences. When a sentence S is learnt as an index to an XP, the concept (for e.g., driving) described in S with appropriate links to other concepts (for e.g., fast, speed) is generalized and installed as a new index to the XP.

In general, there may be more than one deferred sentences before the relevant XP is triggered. Then the issue is, which one of the preceding sentences should be used to form indices to the triggered schema? Let the conceptual representations of the preceding sentences be denoted by S_1, S_2, ..., S_{n-1}, and let S_n represent the current sentence. Assume S_n matches an index to a known XP, allowing the system to retrieve an XP at this point in the story.

initialize set of deferred sentences $\sigma = \{\}$

initialize set of inferred facts $\iota = \{\}$

initialize set of active schemas $\xi = \{\}$

initialize set of active islands $\delta = \{\}$

for each sentence (represented as S) in the story input:

1. if there is already an active schema $XP \in \xi$ that explains S, then use XP; else
2. if S matches an index for a schema $XP \notin \xi$, then
   - activate XP and add it to ξ, and
   - use XP to explain S, and
   - if S can be connected to the last sentence of the closest connected island $d \in \delta$, then learn the first sentence in d as a new index to XP, and add S to d;

   else
3. draw inferences from S and add them to ι. If any of the inferred facts match an index for a schema XP, then:
   - activate XP and add it to ξ, and
   - use XP to explain S, and
   - learn S as a new index to XP;

   else
4. if S matches one of the previous inferences in ι drawn from an earlier sentence S', then S is already connected to S'; else
5. try to connect this sentence to the previous sentence S'. If any of the inferred facts match an index for a schema XP, then:
   - activate XP and add it to ξ, and
   - use XP to explain S, and
   - learn S' as a new index to XP, and
   - try to connect S to the last sentence of the closest connected island $d \in \delta$, and
   - learn the first sentence of the island d, such that S' is in d, as a new index to XP, and add S to d;

   else
6. defer the sentence S and add it to σ. Start a new island $d = \{S\}$ and add it to δ.

repeat until story is over

Figure 1: **The integrated method of schema selection and index learning.**

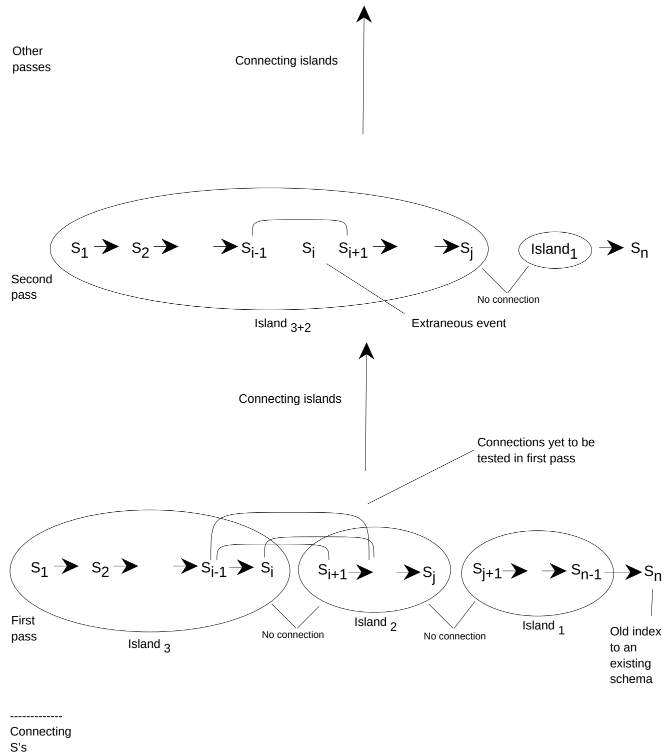

Figure 2: **Connecting conceptual representations and forming islands.**

Some of the S_i's ($i < n$) may be extraneous in the sense that they break the connecting chain from S_1 to S_{n-1} (see figure 2). Typically, there will be broken segments of the chain S_1 to S_{n-1}, which we call *islands*. Islands represent groups of sentences that have been causally connected to each other, but not to other sentences outside the group.[1] Since relevant and connected events are usually described in sequence together in any story, we would expect the island closest to S_n (the first event to match a known index) to contain the appropriate new index that we want the system to learn. Furthermore, the first[2] sentence of any well connected causal chain would be a good index because it predicts subsequent events in that causal chain. Thus a good heuristic for learning indices is to use a generalization of the first sentence of the closest island as the new index.

This heuristic exploits the nature of stories to avoid the need for detailed and complete inferencing which would otherwise be needed to find the right indices. Apart from its intuitive appeal, this method can be very effectively implemented in parallel (in constant time!). The algorithm is:

forall $i := 1$ to $n - 1$ **do**
  connect S_i to S_{i+1}
end

With a second pass over the results of trying to connect S_i and S_{i+1} ($1 \leq i < n$), the islands can be formed sequentially in linear time.

[1] Thus islands are good candidates for generalization into new XPs. When a new island is created, the island being used most recently can be used to learn a new XP, with the first sentence in the island being used to create an initial index to the XP. This is beyond the scope of this paper.

[2] This is the sentence that is first in the causal chain, not necessarily in the temporal order in which the sentences are presented in the story. The earliest sentence may also be a good index if the story is indicative of the typical order in which stories tend to be presented.

An alternate method, similar to back propagation or goal regression (Mitchell et al. 1986; Mitchell et al. 1983; Nilsson 1980) is to try to connect every S_i to S_{i-1} starting at the old index (which matches S_n) until S_1 if possible or propagation fails. At this failure point, say, S_k, the system can use S_k to form a new index to the XP, because it is the earliest sentence that is causally connected to the old index (and hence to the XP).

In general, the problem of connecting S_n with S_k through back propagation may be complicated by the presense of extraneous events. Consider two adjacent islands, say $S_1 \ldots S_j$ and $S_{j+1} \ldots S_k$. Even if S_j and S_{j+1} are not connectable through a known inference rule, the islands may still be connectable in the case that one of S_j or S_{j+1} is an extraneous event. In this situation, if the system tries to connect S_j and S_{j+2}, or S_{j-1} and S_{j+1}, etc., it may be successful in connecting the two islands. In general, there would be several phases of connecting islands, as illustrated in figure 2, that would eventually lead to the most effective chain in a reasonably efficient way.

An alternative to using multiple phases is to hypothesize rules to fill gaps in the causal chains. Whenever there is a gap between events in the story, we hypothesize a rule connecting the end events at the break point (S_j and S_{j+1} in the above example). This relies on the heuristic that these events are probably related since they occur together in the story. However, since this can not be guaranteed to be the case, we would like to keep such hypothetically learned rules or explanations separate from the other explanations until some number of such instances are encountered, after which they can be transferred from the set of hypothetical rules to the fully asserted. This method in effect falls back on similarity based learning to build hypotheses when explanations are not available to guide learning. For instance, in the "driving rashly" and XP-VEHICLE-ACCIDENT example, we could hypothesize that "driving rashly" might lead to an "accident," if none of the known explanations help connect them. Nonetheless, we would assert that as a "fact" only after several incidents are encountered or when an explanation is found. These issues are beyond the scope of this paper.

5 Implementation

We are developing two implementations of our learning methods using different schema-based understanding programs as the base. The first implementation is built on top of the McSAM program (Schank and Riesbeck 1981), which uses scripts (Cullingford 1978; Schank and Abelson 1977) and causal rules (Rieger 1975) to understand natural language stories.

The second implementation is built on top of the explanation component of the AQUA program (Ram 1989; Ram 1990a), which uses explanation patterns to understand newspaper stories.[3] The program starts with three explanation patterns describing typical vehicle accident scenarios: XP-VEHICLE-ACCIDENT-DRUNKEN-DRIVING, in which the accident is caused by the driver's lack of control due to his or her being drunk; XP-VEHICLE-ACCIDENT-RASH-DRIVING (see figure 3), in which the driver is in control of his or her decisions, and makes a decision to drive fast or rashly, thus causing an accident; and XP-VEHICLE-ACCIDENT-PHYSICAL-FAILURE, in which the accident is caused by a physical or mechanical failure of the vehicle. The first two XPs are volitional (Ram 1990a), and refer to the planning and decision processes of the agent. The third XP is physical, and describes the physical causality underlying the accident.

The programs read conceptual representations of simple natural language stories, and use the integrated method of schema selection and index learning shown in figure 1 to understand and learn from these stories. The first implementation has been completed, and the second implementation is currently in progress. We are also developing representations for additional explanation patterns in order to evaluate the generality of the learning algorithm.

[3] Since our implementation focuses on learning issues, we have avoided issues related to parsing by hand-translating the stories into a conceptual representation similar to that of AQUA (Ram 1989).

```
(define-frame XP-VEHICLE-ACCIDENT-RASH-DRIVING
   (isa (list. volitional-xp))
   (explains [collide (actor =agent)
                      (object1 [car =car])
                      (object2 [physical-object
                                  (size [large.0])])
                      (main-result [injured-state
                                      =violated-state
                                      (domain =agent)])
                      (side-effect [destroyed-state
                                      =bad-state
                                      (domain =car)])])
   ...
   (belief-state [drive-fast-to-reach-early-state
                    (actor =agent)
                    (object
                       [results
                          (domain [drive-fast
                                     (actor =agent)])
                          (co-domain [injured-state
                                        (object =agent)])])
                    (belief-strength [very-low.0]) ])
   (links (list. [emotionally-initiates
                    (domain =belief-state)
                    (co-domain =drive-fast)]
                 ...
                 [results
                    (domain =drive-fast)
                    (co-domain =injured-state)]
                  ...
                 [results
                    (domain =injured-state)
                    (co-domain =at-hospital)
                    (truth [in])])) )
```

Figure 3: **The XP-VEHICLE-ACCIDENT-RASH-DRIVING explanation pattern.**

6 Concluding remarks

Although we hope that our theory of index learning for schema selection is applicable to a wide variety of schema-based reasoning tasks, there is little evidence at present that task independent index learning methods are even possible. It may be that indexing is in general task dependent since, even if two different tasks require the same kind of knowledge, the situations in which they require that knowledge could be quite different. Furthermore, each task may provide different cues for retrieval of that knowledge. In order to create a set of indices that are applicable to a variety of different tasks, we need to choose indices based on the functional requirements of these tasks. Functional criteria for index selection can then be used with task independent learning mechanisms (such as explanation-based learning) to learn new indices. To the extent that different tasks (such as planning, prediction, understanding, classification and diagnosis) impose similar functional constraints on the knowledge base or memory system, our methods will yield indices that are task independent. In this paper, we use story understanding as the reasoning task to illustrate our methods. More research is needed to evaluate the applicability of these methods to other reasoning tasks, and thereby to (empirically) address the question of the generality of the methods.

Acknowledgements

We thank reviewers of the paper and Mike Redmond for their comments that helped in improving the presentation. We thank Eric Jones for providing us with his frame system we use in the second implementation. We also thank Mike Cox and Mahesh Kavi for their help in porting the frame system and parts of AQUA.

References

Barletta, R., and Mark, W. 1988. "Explanation-Based Indexing." In *Proceedings of Workshop on Case-Based Reasoning,* 50-60.

Charniak, E. 1978. "With a spoon in hand this must be the eating frame." *Theoretical Issues in Natural Language Processing - 2.* D.Waltz, General Chairman. Univ. of Illinois, Urbana, IL.

Cullingford,R. 1978. "Script application: computer understanding of newspaper stories." Ph.D. thesis, Computer Science Department Research Report 116, Yale University, New Haven, CT.

DeJong, G.F. 1979. "Skimming Stories in Real Time: An Experiment in Integrated Understanding." Computer Science Department Research Report 158, Yale University, New Haven, CT.

DeJong, G., and Mooney, R. 1986. "Explanation-Based Learning: An Alternative View." *Machine Learning,* 1(2):145-176.

Lebowitz, M. 1980. "Generalization and Memory in an Integrated Understanding System." Computer Science Department Research Report 186, Yale University, New Haven, CT.

Mitchell, T.M., Keller, R.M., and Kedar-Cabelli, S.T. 1986. "Explanation-Based Generalization: An unifying view." *Machine Learning,* 1:47-80.

Mitchell, T.M., Utgoff, P., and Banerji, R. 1983. "Learning by Experimentation: Acquiring and Refining Problem-Solving Heuristics." In *Machine Learning: An Artificial Intelligence Approach,* Michalski et al., (Eds.).

Nilsson 1980. *Principles of Artificial Intelligence,* Tioga, Palo Alto, CA.

Ram, A. 1989. "Question-driven Understanding: An integrated theory of story understanding, memory and learning." PhD Thesis, Research Report 710, Yale University, New Haven, CT (May).

Ram, A. 1990a. "Decision Models: A theory of volitional explanation." In *Proceedings of Twelfth Annual Conference of the Cognitive Science Society,* Cambridge, MA, (July): 198-205.

Ram, A. 1990b. "Incremental Learning of Explanation Patterns and their Indices." In *Proceedings of Seventh International Conference on Machine Learning,* Austin, TX, (June): 313-320.

Rieger, C. 1975. "Conceptual Memory and Inference." In *Conceptual Information Processing,* R. Schank (Ed.), North Holland, Amsterdam: 157-288.

Schank, R.C. 1986. *Explanation Patterns: Understanding Mechanically and Creatively.* Lawrence Erlbaum Associates, Hillsdale, NJ.

Schank, R.C., and Abelson, R. 1977. *Scripts, Plans, Goals and Understanding: An Inquiry into Human Knowledge Structures.* Lawrence Erlbaum Associates, Hillsdale, NJ.

Schank, R.C., and Riesbeck, C.K. 1981. *Inside Computer Understanding: Five Programs Plus Miniatures.* Lawrence Erlbaum Associates, Hillsdale, NJ.