AI Magazine Volume 12 Number 1 (1991) (© AAAI)

Case-Based Reasoning: A Research Paradigm

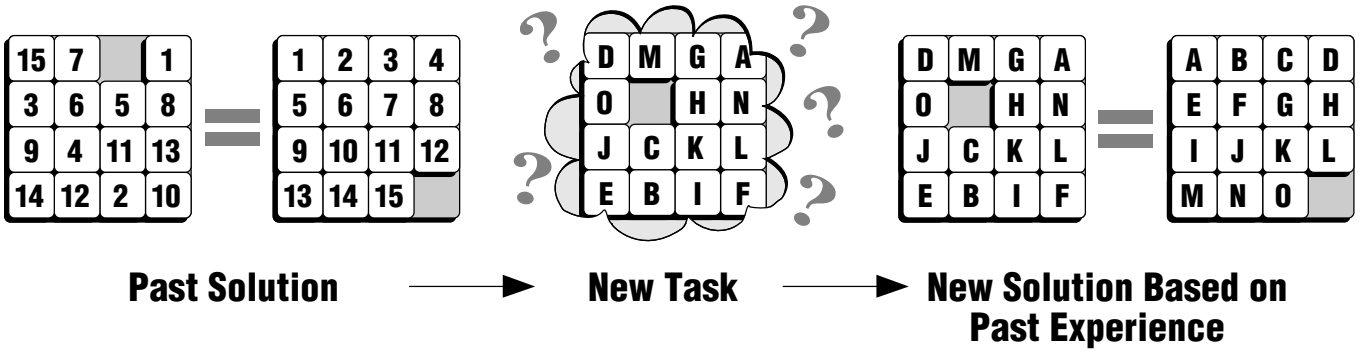
Stephen Slade

Expertise comprises experience. In solving a new problem, we rely on past episodes. We need to remember what plans succeed and what plans fail. We need to know how to modify an old plan to fit a new situation. Case-based reasoning is a general paradigm for reasoning from experience. It assumes a memory model for representing, indexing, and organizing past cases and a process model for retrieving and modifying old cases and assimilating new ones. Case-based reasoning provides a scientific cognitive model. The research issues for case-based reasoning include the representation of episodic knowledge, memory organization, indexing, case modification, and learning. In addition, computer implementations of case-based reasoning address many of the technological shortcomings of standard rule-based expert systems. These engineering concerns include knowledge acquisition and robustness. In this article, I review the history of case-based reasoning, including research conducted at the Yale AI Project and elsewhere.

"I have but one lamp by which my feet are guided, and that is the lamp of experience. I know no way of judging of the future but by the past."
—*Patrick Henry*
Speech in Virginia Convention, Richmond
March 23, 1775

There are two broad research agendas in AI. The first is scientific. AI researchers seek to understand the nature of intelligence and human thought. They examine a range of human cognitive behavior, including memory, learning, planning, and problem solving and look for principles that play general descriptive and explanatory roles. AI shares these scientific ambitions with other cognitive science disciplines. The second agenda for AI research is technological. Researchers seek to create intelligent artifacts, machines that can perform useful tasks. They want to develop the technology of intelligence. They want to be able to design and build computer programs that can solve problems and adapt to new situations.

In this article, I discuss case-based reasoning, an AI paradigm that addresses both research agendas. *Case-based reasoning* is a psychological theory of human cognition. It addresses issues in memory, learning, planning, and problem solving. Case-based reasoning also provides a foundation for a new technology of intelligent computer systems that can solve problems and adapt to new situations. I first review the underlying psychological model of case-based reasoning. I then examine several



0738-4602/90/$4.00 ©1990 AAAI

computer models that embody the principles of case-based reasoning, contrasting the case-based approach with the rule-based expert system paradigm.

Models of Memory

An intelligent being requires knowledge about the world. Knowledge allows a person to plan and solve problems. Knowledge is a resource, a commodity. Memory is the repository of knowledge. The question for the psychologist has been what theory of memory accounts for observed cognitive behaviors. The question for the AI researcher has been how to represent knowledge in a computer program and apply it to perform specific tasks.

Semantic and Episodic Memory

These questions converged as AI researchers attempted to create computer programs that model cognitive processes. A leading theory has been the semantic network memory model. Psychologists have devoted much attention to this theory (Collins and Quillian 1969; Rumelhart, Lindsay, and Norman 1972; Kintsch 1972) as have AI researchers (Quillian 1968; Woods 1975).

Semantic networks typically represent static facts about the world, such as Fido is a dog, a dog is a mammal, and mammals have hair. In general, this type of knowledge does not change over time. Psychologists and AI researchers realized that semantic networks did not account for all the data. First, not all knowledge is in small, static chunks. Memories are variable in size and malleable in content. For example, the memory of what I had for lunch today might vary from nothing to a seven-course meal. Second, the semantic network theory does not explain how knowledge is incorporated into memory. Where does the information come from? It is clear that we are not born with an innate knowledge of the world. At the least, we did not arrive here with advanced knowledge of all our luncheon menus.

To address these and other questions, Tulving (1972, 1983) proposed a theory of episodic memory as an adjunct to semantic memory. Tulving described semantic and episodic memory as two complementary information-processing systems, both of which receive information from perceptual and cognitive systems, process portions of the information, and communicate information to other behavioral and cognitive systems.

Semantic and episodic memory, according to Tulving, differ in the type of information stored, autobiographical reference versus cognitive reference, retrieval conditions and consequences, the volatility of stored information; and interdependence.

More specifically,

> Episodic memory receives and stores information about temporally dated episodes or events, and temporal-spatial relations among these events. A perceptual event can be stored in the episodic system solely in terms of its perceptible properties or attributes, and it is always stored in terms of its autobiographical reference to the already existing contents of the episodic memory store. The act of retrieval of information from the episodic store, in addition to making the retrieval contents accessible to inspection, also serves as a special type of input into episodic memory and thus changes the contents of the episodic memory store. (Tulving 1972, pp. 385–386)

Episodic memory provides an account of representing and recalling larger chunks of temporally related information—events, scenes, occurrences, and stories. By contrast, "semantic memory is the memory necessary for the use of language. It is mental thesaurus" (Tulving 1972, p. 386).

Conceptual Memory

In parallel with the identification of episodic memory by psychologists, AI researchers arrived at a similar theory for language-understanding tasks. Schank (1972, 1975a) and his students developed natural language systems for representing concepts and understanding single sentences. A sentence such as "John ate a hamburger" could be processed, paraphrased, and translated to another language. The next step was to process connected text—paragraphs and stories. For this task, Schank (1975b) proposed a conceptual memory that combined semantic memory with Tulving's episodic memory:

> The distinction between semantic memory and episodic memory is a false one. We shall argue that what must be present is a lexical memory which contains all of the information about words, idioms, common expressions etc., and which links these to nodes in a conceptual memory, which is language free. We believe that it is semantic memory rather than episodic memory which is the misleading notion. Once we change semantic memory by separating out lexical memory, we are left with a set of associations and other relations between con-

Case-based reasoning is a psychological theory of human cognition.

Scripts were proposed as a knowledge structure for a conceptual memory.

> cepts that could only have been acquired by personal experience. We claim that conceptual memory, therefore, is episodic in nature. (Schank 1975b, pp. 255–256)

A key feature of Schank's conceptual memory is the notion that information is derived from experience. Knowledge is not innate. A theory of memory must account for the acquisition of knowledge.

Scripts, Memory Organization Packets, and Reminding

Schank and Abelson (1975, 1977) proposed knowledge structures for representing episodic information. Their primary knowledge structure was the script. Scripts accounted for information about stereotypical events, such as going to a restaurant, taking a bus, or visiting the dentist. In such common situations, a person has a set of expectations concerning the default setting, goals, props, and behaviors of the other people involved. Scripts are analogous to Minsky's (1975) frames, which were proposed in the context of visual processing. It is important to note that scripts are directly related to autobiographical events. Scripts are inherently episodic in origin and use. That is, scripts arise from experience and are applied to understand new events.

Scripts were proposed as a knowledge structure for a conceptual memory. The acquisition of scripts is the result of repeated exposure to a given situation. For example, children learn the restaurant script by going to restaurants over and over again. As a psychological theory of memory, scripts suggested that people would remember an event in terms of its associated script. However, an experiment by Bower, Black, and Turner (1979) showed that subjects often confused events that had similar scripts. For example, a subject might mix up waiting room scenes from a visit to a doctor's office with a visit to a dentist's office.

These data required a revision in script theory. What knowledge structures would allow for such confusion? What was the underlying process of remembering?

Schank (1979, 1980, 1982) postulated a more general knowledge structure to account for the diverse and heterogeneous nature of episodic knowledge. This new structure was the memory organization packet (MOP). MOPs can be viewed as metascripts. For example, instead of a dentist script or a doctor script, there might be a professional-office-visit MOP that can be instantiated and specified for both the doctor and the dentist. This MOP would contain a generic waiting room scene, thus providing the basis for confusion between doctor and dentist episodes.

More important than the MOP knowledge structure was the new emphasis on the basic memory processes of reminding and learning. The early work of Bartlett (1932) on remembering influenced the original design of scripts for story comprehension (Schank 1975b). The new focus on reminding raised additional questions about how memory was organized and indexed. Schank illustrated this phenomenon with sample remindings, such as the following, which were informally gathered from colleagues and students:

> *The steak and the haircut.*
>
> X described how his wife would never make his steak as rare as he liked it. When this was told to Y, it reminded Y of a time, 30 years earlier, when he tried to get his hair cut in a short style in England, and the barber would not cut it as short as he wanted it.
>
> *The sand dollars and the drunk.*
>
> X's daughter was diving for sand dollars. X pointed out where there were a great many sand dollars, but X's daughter continued to dive where she was. X asked why. She said that the water was shallower where she was diving. This reminded X of the joke about the drunk who was searching for his ring under the lamppost because the light was better there even though he had lost the ring elsewhere. (Schank 1982, p. 47)

The remindings in these two stories have little to do with the surface features of the episodes. Cooking a steak and cutting hair are hardly similar events, yet the stories are related. These examples illustrate the fact that memory has a complex structure and indexing that allows people to relate a new episode to prior cases through thematic and abstract categories.

Given this richly indexed structure, Schank proposed a theory of learning based on reminding. If we assume that new situations or experiences will remind us of previous cases and events, we can classify a new

episode in terms of past cases. The knowledge of the past case, like a script, can guide our behavior. We can rely on the past episode to help us understand a new situation. For example, the second time we ride in an airplane, we will be reminded of our first airplane trip. We can use this experience to remind us to get a boarding pass, find our seat, stow our luggage, fasten the seatbelt, take some Dramamine, and so on.

However, when the new situation does not conform to the prior case, we have a failure. That is, we had an expectation based on a prior event that did not occur in the new situation. Thus, we must classify this new situation as different from the previous episode. We must remember this new experience. We must learn. Schank (1981) termed this process *failure-driven learning*. Returning to the airplane example, if we have flown several times but then take the air shuttle for the first time, we will have some surprises. The expectations of having an assigned seat and buying our ticket ahead of time fail. We must recognize these discrepancies and account for them. We must modify our knowledge structure for airplane rides so that the next time we take the air shuttle, we will know better what to expect.

When we observe a discrepancy between our predictions and some event, we have something to learn. We need to revise our knowledge structure. The mechanism for updating our knowledge often requires explanation. Schank (1986) noted that explanation plays a central role in learning and intelligence. He proposed an explicit knowledge structure, *explanation patterns* (XPs), that is used to generate, index, and test explanations in conjunction with an episodic memory.

Process Model

I began with the intent of representing knowledge, thus deriving a theory of memory to account for episodic information. Scripts and MOPs were postulated as knowledge structures for representing experience. However, the knowledge structures provide only part of the answer. We must also specify the processes involved in acquiring and accessing these structures. We need a process model.

In figure 1 (after Riesbeck and Bain [1987]), I present a flowchart that illustrates the basic process of case-based reasoning and learning. Boxes represent processes, and ovals represent knowledge structures. The process of interpreting and assimilating a new event breaks down into the following steps, starting with an input event, as shown at the top of the flowchart:

1. Assign Indexes: Features of the new event are assigned as indexes characterizing the event. For example, our first air shuttle flight might be characterized as an airplane flight.

2. Retrieve: The indexes are used to retrieve a similar past case from memory. The past case contains the prior solution. In our example, we might be reminded of a previous airplane trip.

3. Modify: The old solution is modified to conform to the new situation, resulting in a proposed solution. For our airplane case, we would make appropriate modifications to account for changes in various features such as destination, price, purpose of the trip, departure and arrival times, weather, and so on.

4. Test: The proposed solution is tried out. It either succeeds or fails. Our airplane reminding generates certain expectations, not all of which can be met.

5. Assign and Store: If the solution succeeds, then assign indexes and store a working solution. The successful plan is then incorporated into the case memory. For a typical airplane trip, there will be few expectation failures and, therefore, little to make this new trip memorable. It will be just one more instance of the airplane script.

6. Explain, Repair, and Test: If the solution fails, then explain the failure, repair the working solution, and test again. The explanation process identifies the source of the problem. The predictive features of the problem are incorporated into the indexing rules to anticipate this problem in the future. The failed plan is repaired to fix the problem, and the revised solution is then tested. For our air shuttle example, we realize that certain expectations fail. We learn that we do not get an assigned seat and that we do not have to pay ahead of time. We might decide that taking the air shuttle is more like riding on a train. We can then create a new case in memory to handle this new situation and identify predictive features so that we will be reminded of this episode the next time we take the shuttle.

In support of this process are the following types of knowledge structures, represented by ovals in the figure:

Indexing Rules: *Indexing rules* identify the predictive features in the input that provide appropriate indexes into the case memory. Determining the significant input features is a persistent problem (Schank, Collins, and Hunter 1986).

Case Memory: *Case memory* is the episodic memory, which comprises the database of experience.

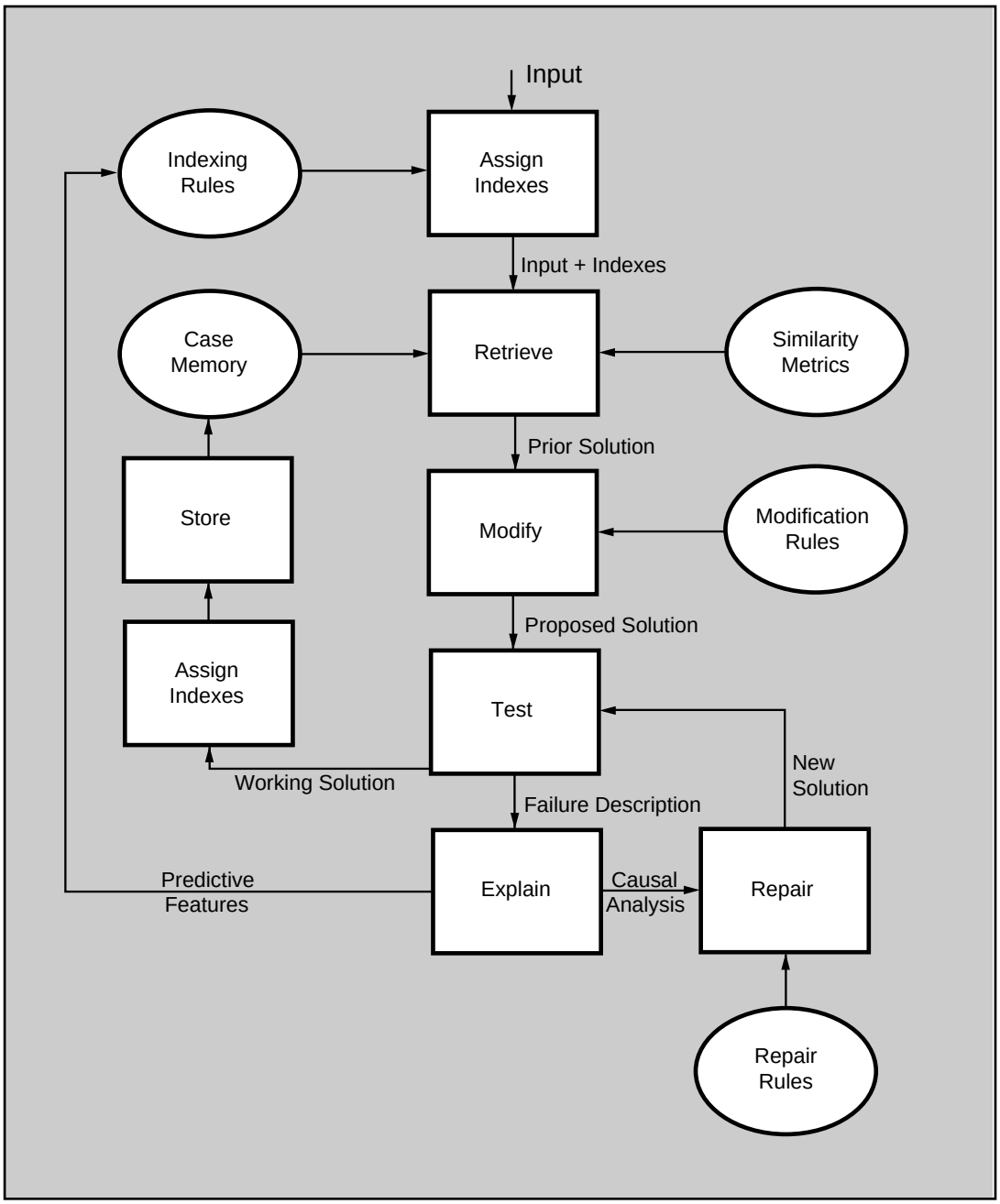


Figure 1. Case-Based Reasoning Flowchart.

Similarity Metrics: If more than one case is retrieved from episodic memory, the *similarity metrics* can be used to decide which case is more like the current situation. For example, in the air shuttle case, we might be reminded of both airplane rides and train rides. The similarity rules might initially suggest that we rely on the airplane case.

Modification Rules: No old case is going to be an exact match for a new situation. The old case must be modified to fit. We require knowledge about what kinds of factors can be changed and how to change them. For the airplane ride, it is acceptable to ride in a different seat, but it is usually not advisable to change roles from passenger to pilot.

Repair Rules: Once we identify and explain an expectation failure, we must try to alter our plan to fit the new situation. Again, we have rules for what kinds of changes are permissible. For the air shuttle, we recognize that paying for the ticket on the plane is an acceptable

change. We can generate an explanation that recognizes an airplane ride as a type of commercial transaction and suggests that there are alternative acceptable means of paying for services.

Psychological Issues

The process model depicted in figure 1 is not meant to stipulate the necessary and sufficient conditions for simulating cognitive behavior. Rather, it illustrates a variety of salient issues in case-based reasoning.

I can summarize the psychological assumptions of the case-based reasoning paradigm as follows:

First, memory is predominantly episodic. The primary content of memory is experience.

Second, memory is richly indexed. Experiences are related to each other in many complex and abstract ways.

Third, memory is dynamic. The organization and structure of memory changes over time.

Fourth, experience guides reasoning. We interpret and understand new situations in terms of prior experience.

Fifth, learning is triggered by failure. When an expectation from a previous case fails to predict a new situation, we learn by incorporating the new episode into memory.

Similarly, we can present the research questions that arise from these respective assumptions:

First, what makes up a case? What is the content and structure of an episode in memory? What is the relationship between episodic memory and other types of knowledge? How can we represent case memory?

Second, how is memory organized? What set of indexes is appropriate for classifying cases? What search algorithms complement the structure of memory? What are the indexing rules?

Third, how does memory change? What leads to forgetting? How does the memory of cases and stories degrade? How do the case memory and indexing rules change over time?

Fourth, how can we adapt old solutions to new problems? How can we recognize a new situation as similar to a previous episode? What are the similarity metrics and modification rules?

Fifth, what leads us to reject or accept a new case that is in conflict with a previous case? How do we explain the differences between episodes? How can we learn from mistakes? What are the repair rules?

It might seem that I present more questions than answers. However, my basic premise is that case-based reasoning provides a foundation for a broad range of research. It is appropriate and, indeed, desirable to stimulate research through the principled identification and examination of cognitive phenomena. I now review the history of case-based reasoning in AI research.

Computer Models

Many of the principles of case-based reasoning can be found in Sussman's (1975) HACKER program. HACKER's answer library was similar to a case memory, and its debugging process was analogous to plan repair. Furthermore, the underlying cognitive premise of the HACKER model was learning through experience, clearly at the heart of the case-based reasoning paradigm. The episodes for HACKER were restricted to computer programs rather than more general human experiences.

The first computer programs to use scripts were SAM (script applier mechanism) (Cullingford 1978) and FRUMP (fast reading understanding memory program) (DeJong 1979). These programs read newspaper stories and performed various language tasks, such as translation, summarization, and question answering. These programs contained static knowledge structures that were used in processing stories. The content of the programs' memory did not change as a result of processing—in spite of the memory in FRUMP's name.

These programs were a successful demonstration of the natural language processing of stories and of scripts as a knowledge structure. Understanding a story entailed processing an episode or event. Scripts provided a feasible means for representing such episodic knowledge. However, the programs failed to demonstrate knowledge acquisition. The scripts of SAM and FRUMP were innate, as it were, having been written by programmers. The programs used the scripts to guide the processing of stories, but the programs did not learn their scripts through experience.

Furthermore, the programs did not remember anything. SAM and FRUMP could read the same story 20 times in a row and not recognize that they previously saw this story. Clearly, a program that modeled human memory should remember its own experience.

Two programs that addressed the issue of memory organization for episodic knowledge were CYRUS and IPP. CYRUS (Kolodner 1980; Schank and Kolodner 1979; Kolodner 1984; Kolodner and Cullingford 1986) simulated an episodic memory of events relating to former Secretary of State Cyrus Vance. The program answered questions about a range of autobiographical episodes, such as meetings, diplomatic trips, and state dinners. CYRUS was the

The programs from the late 1970s that modeled episodic memory were largely natural language-processing programs.

first program to model episodic storage and retrieval strategies. Although the focus of CYRUS was on memory organization and indexing, an attempt was made to integrate CYRUS with the FRUMP newswire program to provide an automatic update for CYRUS's memory (Schank, Kolodner, and DeJong 1980). The combined system, Cyfr, read news stories about the secretary of state and integrated the events into CYRUS's episodic memory.

IPP (Lebowitz 1980) was a prototype case-based reasoning and learning program. IPP read news stories about terrorist acts, such as bombings, kidnappings, and shootings. The program started with generic knowledge about such acts and, after reading hundreds of stories, developed its own set of generalizations about terrorism that it could apply to understanding new stories. For example, IPP read the following two stories about terrorism by the Irish Republican Army (IRA) in Northern Ireland:

> Story: XX1 (4 12 79) Northern-Ireland (Irish Republican Army guerillas ambushed a military patrol in West Belfast yesterday killing one British soldier and badly wounding another Army headquarters reported)
>
> Story: XX2 (11 11 79) Northern-Ireland (A suspected Irish Republican Army gunman killed a 50-year-old unarmed security guard in East Belfast early today the police said)

The program noticed that in each case, the victims were establishment, authority figures (policemen and soldiers) and that the terrorists were IRA members. IPP formed a generalization based on this similarity. It then read the following story about another shooting in Northern Ireland:

> Story: XX3 (1 12 80) Northern-Ireland (A gunman shot and killed a part-time policeman at a soccer match Saturday and escaped through the crowd to a waiting getaway car ^comma^ police said.

Based on its prior experience, IPP inferred that the unidentified gunman in story XX3 is an IRA member. This inference might or might not be correct, but it demonstrates the ability to relate previous episodes to new situations.

Expert Systems: Rules versus Cases

The programs from the late 1970s that modeled episodic memory were largely natural language–processing programs. By this time, another topic of AI research had developed into a primary application area, namely, rule-based expert systems. Early programs such as DENDRAL (Buchanan, Sutherland, and Feigenbaum 1969) and MYCIN (Shortliffe 1976) demonstrated the possibility of simulating the problem-solving ability of human experts, such as chemists or physicians. The success of these and other programs stimulated interest in developing expert systems for a vast number of technical applications.

The basic unit of knowledge in these expert systems was the rule. A rule comprised a conditional test-action pair, for example, if condition, then action. Several hundred rules might be required for a typical diagnostic or repair task.

Building rule-based or production systems became a popular enterprise. As experience with expert systems increased, so did awareness of some basic shortcomings of the rule-based system paradigm.

The first problem was knowledge acquisition. To build an expert system, a computer programmer (or knowledge engineer) had to sit with the human expert to determine what rules were appropriate for the given domain. This knowledge was difficult to uncover. The human expert could not simply make a list of the hundreds of rules s/he used to solve problems. Often, the informant articulated a set of rules that in fact did not accurately reflect his(her) own problem-solving behavior. For these reasons, this difficult knowledge-acquisition process became known as a bottleneck in constructing rule-based expert systems (Hayes-Roth, Waterman, and Lenat 1983).

Second, the rule-based systems did not have a memory. That is, just as SAM and FRUMP would not remember news stories that they had already read, rule-based systems would not remember problems that they had already solved. For example, if a medical diagnosis program were presented with a patient with a certain set of symptoms, the program might have fired dozens or hun-

dreds or thousands of rules to come up with a diagnosis or treatment. Subsequently, if the program were presented with another patient displaying the same set of symptoms, the program fired the same set of rules. The program did not remember having previously seen a similar patient. One might state that this observation is of little consequence beyond some argument for computational efficiency. However, efficiency can be a significant concern in many situations. Moreover, a program without a memory cannot remember its mistakes and, thus, is destined to repeat them. Thus, accuracy and efficiency are related problems for a system without a memory.

Third, rule-based systems were not robust. If a problem were presented to the system that did not match any of the rules, the program could not respond. The system's knowledge base was limited to its rules, so if none of the rules could apply, the system had no alternatives. It was brittle.

We can compare the behavior of the rule-based expert system with the behavior of the human expert. First, the central feature of expertise is experience. An expert is someone who has vast, specialized experience; has witnessed numerous cases in the domain; and has generalized this experience to apply it to new situations. When confronted with a problem, the expert is reminded of previous, similar problems and their respective resolutions. It might be that the expert has so many exemplars for a given problem that the experiences have been distilled into a general rule to be applied. Still, this general rule has its roots in actual experience.

Thus, the human expert derives knowledge from experience. The basic unit of knowledge is not the rule but the case. Human experts acquire knowledge by assimilating new cases, either first hand or through reports from others. Furthermore, it is easier for people to articulate knowledge as experience than rules. This observation suggests the psychological hypothesis that expert knowledge might, in fact, primarily be encoded as episodes rather than rules. Contrast this acquisition of knowledge from experience with the knowledge-acquisition bottleneck given as the first problem of rule-based systems.

Second, human experts remember their own experience. The doctor who fails to effectively treat a case should remember this case when another patient presents the same symptoms. The doctor can learn from his(her) mistakes.

Third, human experts can reason by analogy. If our doctor sees a patient who presents symptoms that are unlike anything in his(her) experience, the doctor does not need to simply give up. The doctor might be reminded of various previous cases that were similar in one way or another and devise a treatment accordingly. Just as our first air shuttle trip might remind of us of both an airplane trip and a train ride, the doctor might be able to arrive at a composite diagnosis based on different earlier cases.

These arguments suggest an alternative to the rule-based system: a case-based system. An expert system that can extract information from its experience can grow and acquire knowledge on its own. This ability is crucial for the long-range success of the expert system concept in AI. The automated reasoning power can be applied to so many tasks that it is necessary to develop a mechanism that can directly assimilate new knowledge from experience.

The technology of case-based systems directly addresses problems found in rule-based systems: First is knowledge acquisition. The unit of knowledge is the case, not the rule. It is easier to articulate, examine, and evaluate cases than rules. Second is performance experience. A case-based system can remember its own performance and modify its behavior to avoid repeating prior mistakes. Third are adaptive solutions. By reasoning from analogy with past cases, a case-based system should be able to construct solutions to novel problems.

The scientific research issues previously given for case-based reasoning models also directly apply to the technological research issues for case-based systems. We must answer these same questions in building case-based systems:

First, what makes up a case? How can we represent case memory?

Second, how is memory organized? What are the indexing rules?

Third, how does memory change? How do the case memory and indexing rules change over time?

Fourth, how can we adapt old solutions to new problems? What are the similarity metrics and modification rules?

Fifth, how can we learn from mistakes? What are the repair rules?

At this point, the astute reader might ask why case-based systems use rules for indexing, modification, and repair because rules seem to be at the heart of so many problems with rule-based systems. There are two answers. First, the rules in case-based systems do not make up the primary knowledge base but, rather, independent support modules. Thus, there should be less complexity. However, the theory of case-based reasoning suggests that these rules would themselves be acquired by experience from cases through a recursive

CYRUS and IPP can be viewed as prototypes for case-based systems.

CHEF demonstrates how episodic knowledge can be used to guide planning and avoid past failures.

application of the case-based reasoning algorithm. That is, the system would derive rules for indexing, modification, and repair from cases and experience. Early case-based systems have not addressed this problem, but the basic paradigm suggests this approach.

The technology of case-based systems is an instantiation of the psychological theories of case-based reasoning. CYRUS and IPP can be viewed as prototypes for case-based systems. They began to address the fundamental questions of case representation and indexing previously posed. In the 1980s, researchers began explicitly to develop case-based systems.

Case-Based Systems

To see how the technology of case-based systems developed, we first look at Hammond's (1984, 1986, 1989) CHEF program. Unlike CYRUS and IPP, CHEF is not based in a natural language-understanding task but instead focuses on planning. CHEF develops new plans based on its own experience in the domain of cooking. When faced with the task of preparing a dish for which it has no appropriate plan (recipe), CHEF modifies an existing plan to fit the new situation and then tries to detect and correct any errors that result. CHEF learns from its own mistakes.

CHEF demonstrates how episodic knowledge can be used to guide planning and avoid past failures. When presented with a problem—how to prepare a certain dish—the program is reminded of previous related recipes. It modifies the most similar previous recipe to fit the new requirements and then tries out the new recipe. CHEF tests the recipe through a simulation involving rules that specify the physical effects of each step of the cooking process. The results are then examined to see if they match the goals of the intended dish. If the program recognizes a failure, it then tries to analyze and explain the failure through a process of reasoning by asking questions. Finally, the program modifies the recipe in light of its explanation to correct the failure. This case-based planning process closely follows the flowchart in figure 1.

For example, when presented with the task of creating a strawberry souffle, CHEF resorts to modifying a vanilla souffle recipe. However, simply adding strawberries to the standard recipe keeps the souffle from rising properly. CHEF discovers the source of the problem in the excess liquid from the berries and decides that the best remedy is to add more whipped egg whites. This solution fixes the recipe. CHEF never repeats this mistake and can use this experience in other recipes, such as a raspberry souffle.

CHEF provides one set of answers for our cardinal research questions:

First, what makes up a case? For CHEF, cases are recipes—a particular set of plans. CHEF suggests the feasibility of looking at planning as case-based reasoning.

Second, how is memory organized? CHEF's memory is indexed in many ways, including goals, plan failures, and plan interactions.

Third, how does memory change? CHEF learns from mistakes. When a failure occurs, CHEF must identify the source of the failure, fix the plan, and remember the result. CHEF tries not to make the same mistake twice.

Fourth, how can we adapt old solutions to new problems? CHEF can create new recipes by starting with old recipes. In selecting an old plan, CHEF tries to match as many of the new plan's goals as possible. The modification to the recipe is driven first by a rudimentary knowledge of the ingredients and then by trial-and-error testing.

Fifth, how can we learn from mistakes? CHEF can identify errors when it observes that a recipe does not satisfy its intended goals. At this point, CHEF must find the source of the error and apply a set of strategies for modifying the plan. Many of CHEF's modification strategies are domain specific, relating to the substitution or preparation of cooking ingredients.

As an ongoing research enterprise, case-based reasoning finds new ways of asking and addressing these questions. I now review a number of case-based systems in a variety of domains.

Another early case-based expert system is Simpson's MEDIATOR (Kolodner, Simpson, and Sycara-Cyranski 1985; Simpson 1985). The program acts as an advisory system for dispute mediation. MEDIATOR is presented with a dispute situation between two parties and suggests a resolution based on its experiential knowledge base. The program addresses problems of similarity measures, memory structures for representing and retrieving cases, adaptation, and recovery from failure.

For example, when faced with the dispute between Israel and Egypt over control of the

Sinai, MEDIATOR first selects the precedent of the Korean conflict in which the disputed territory was equally divided. This proposal fails in the case of the Sinai because Israel's goal is one of national security, but Egypt is concerned with territorial integrity. In the process of explaining the failure, MEDIATOR is reminded of the Panama Canal treaty, which divided the disputed territory in another manner, ceding economic and political control to Panama, with the United States retaining military control. MEDIATOR then proposes this case as an appropriate precedent for the Sinai dispute.

JUDGE (Bain 1984; Bain 1986) applies the case-based approach to legal reasoning in the context of sentencing convicted criminals. The program simulates the process of a judge deciding the appropriate sentence to mete out based on the judge's experience. The following is a brief example from JUDGE (Bain 1986). The program is given a new case, CRIME8, which has features that remind JUDGE of a previous assault case, CRIME1. JUDGE compares the two cases to decide what sentence should be imposed in the new case. The program explains its reasoning as it goes along:

CRIME8 Facts:

First, Terry struck Sam with his fists several times.

Sam was only slightly hurt.

[Interpreted as unprovoked-violation]

Next, Sam struck Terry with his fists several times.

Terry was only slightly hurt.

[Interpreted as parity-self-defense with an achieved-result]

Finally, Terry struck Sam with his fists several times.

Sam received a concussion.

[Interpreted as parity-retaliation with a violative-accidental-result.]

The only thematically similar previous situation in memory was CRIME1. CRIME1 contained a similar final intention and result interpretation as CRIME8.

Comparing CRIME1 from memory with the current input, CRIME8…

The old crime, CRIME1, once again, involved the following:

First, Randy struck Chuck with his fists several times.

Chuck was only slightly hurt.

[Interpreted as unprovoked-violation]

Next, Chuck struck Randy with his fists several times.

Randy was only slightly hurt.

[Interpreted as parity-self-defense with an achieved-result]

Program	Year	Domain	Task
CYRUS	1980	Diplomacy	Memory organization and retrieval
IPP	1980	Terrorism	Story understanding/learning
MEDIATOR	1985	Mediation	Case-based problem solving
JUDGE	1986	Law	Case-based assessment
CHEF	1986	Cooking	Case-based planning/learning
HYPO	1987	Law	Case-based reasoning

Table 1. Early Case-Based Systems.

Then, Randy slashed at Chuck with a knife one time.

Chuck's skin was slightly cut.

[Interpreted as escalated-retaliation with a bungled-result.]

Next, Chuck slashed at Randy with a knife one time.

Randy's skin was slightly cut.

[Interpreted as parity-self-defense with a bungled-result.]

Finally, Randy stabbed Chuck with a knife several times.

Chuck died.

[Interpreted as escalated-retaliation with an achieved-result]

The result where Chuck was killed was worse than the one in which Sam received a concussion. Not only was the outcome in the newer situation accidental, but Randy both intended and caused more harm than Terry.

The intent of Randy, the offender in the older crime, to act repeatedly to stab Chuck to death was more severe than that of Terry, the other offender, to act repeatedly to hit Sam. In addition, although neither actor's intentions were justified, Randy escalated the level of violence, while Terry did not.

Comparison finished with result that the old crime, CRIME1, was worse.

The sentence originally given for CRIME1, for violation of murder, was a term of imprisonment of not less than 40 years nor more than 50 years to be served in the state's prison. Since the circumstances of CRIME1 are found to be worse than CRIME8, due to more severe actions and results at the end of the crime, the offender will get a sentence which is more lenient than the sentence for CRIME1, relative to the crime of assault in the second degree.

Program	Citation	Site	Domain	Task
PLEXUS	Alterman 1986	Berkeley	Travel	Adaptive planning
CBD	Hammond & Hurwitz 1988	Chicago	Machines	Case-based diagnosis
TRUCKER	Marks et al 1988	Chicago	Scheduling	Pluralistic planning
ROENTGEN	Berger 1989	Chicago	Medicine	Case-based planning
CYCLOPS	Navinchandra 1988	CMU	Landscaping	Design problem solving
PERSUADER	Sycara 1988	CMU	Mediation	Plan adaptation and repair
PRODIGY	Carbonell & Veloso 1988	CMU	Algebra	Derivational analogy
CBR Shell	Riesbeck 1988	CSI	Tool	Programming shell
JULIA	Shin 1988	GA Tech	Cooking	Analogical reasoning
MEDIC	Turner 1988	GA Tech	Medicine	Diagnostic reasoning
PARADYME	Kolodner 1988	GA Tech	Cooking	Parallel memory retrieval
CLAVIER	Mark 1989	Lockheed	Autoclave layout	Case-based reasoning
CASEY	Koton 1988	MIT	Medicine	Reasoning about evidence
SMARTplan	Koton 1989	MITRE	Scheduling	Resource allocation
n/a	Wall et al., 1988	TI	Military	Tactical planning
OGRE	Donahue 1989	TI	Tool	Reasoning shell
JOHNNY	Stanfill 1988	TMC	Reading	Memory-based reasoning
PHI-PSI	Zhang & Waltz 1989	TMC	Protein structure	Memory-based reasoning
LAWCLERK	Selfridge & Cuthill 1989	UConn	Law	Cross-context reminding
CBS	Bradtke & Lehnert 1988	UMass	Puzzles	Case-based search
TA	Williams 1988	UMass	Programming	Case-based learning
CABARET	Skalak 1989	UMass	Tool	Reasoning shell
PROTOS	Bareiss 1988	UTexas	Medicine	Exemplar-based learning
GREBE	Branting 1989	UTexas	Law	Exemplar-based explanation
ANON	Owens 1988	Yale	Proverbs	Indexing prototypical cases
DMAP	Riesbeck & Martin 1985	Yale	Economics	Direct memory access parsing
IVY	Hunter 1989	Yale	Medicine	Case-based diagnosis
SWALE	Kass & Leake 1988	Yale	Post-mortem	Case-based explanations

Table 2. Recent Case-Based Research Projects.

> The sentence to be given for CRIME8, for violation of assault in the second degree as defined in Section 53A-60 in the Connecticut penal code, will be a term of imprisonment of not less than 3 years to be served in the state's prison. The maximum period of incarceration allowed for violation of this statute is 5 years.

This example illustrates some of the main aspects of case-based reasoning. First, JUDGE has a case library comprising an episodic memory of previously adjudicated crimes. Second, new cases are analyzed to provide indexes for retrieving past cases. Third, JUDGE can compare the new case with the old case to determine how the new situation might require modification of the prior sentence. Fourth, JUDGE provides an explanation of its reasoning by explicit analysis and comparison of the two cases. Finally, the new case becomes assimilated into JUDGE's case library for future use.

The HYPO program (Ashley 1987; Rissland and Ashley 1988) is another legal reasoning system. Developed at the University of Massachusetts at Amherst, HYPO analyzes legal problems in domains such as tax law and trade secrets. Given a description of a legal dispute, HYPO, like any good lawyer, can provide arguments, supported by appropriate legal precedents, on both sides of the case. HYPO's case library comprises actual legal cases. A key focus of the HYPO system is assessing the similarity between the problem situation and relevant cases in the knowledge base.

Table 1 summarizes these early programs, all of which represent completed Ph.D. research. Recent work in case-based reasoning has proceeded apace, reflecting a widespread and growing interest in the case-based paradigm.

Just as law seemed a natural domain for early case-based reasoning systems, medicine also proved an attractive subject. Medical education places great emphasis on learning from experience through internships and residencies in which doctors are exposed to a wide range of patients, symptoms, diseases, and treatments. The basic patient episode is a case.

PROTOS (Bareiss 1988; Bareiss, Porter, and Wier 1988), developed at the University of Texas, reasons from cases in the domain of clinical audiology (hearing disorders). PROTOS was trained with 200 clinical cases from the Baylor College of Medicine. A case consists of the symptoms reported by the patient, the patient history, and the results of routine tests. When presented with a new case, PROTOS tries to match it against previous cases to classify the new case and arrive at a diagnosis. Cases can be grouped under exemplars based on how prototypical their features are. In the training set, PROTOS learned 24 diagnostic categories and 120 exemplars. Following training, PROTOS was given a set of 26 test cases on which it scored over 90-percent accuracy.

Recent case-based research projects are summarized in table 2. This list is not exhaustive, merely illustrative. These efforts demonstrate a wide range of domains and research issues that have flowed from the case-based reasoning paradigm.

Summary

In this article, I reviewed some of the beginnings, motivations, and trends in case-based reasoning research. Case-based reasoning grew out of psychological models of episodic memory and the technological impetus of AI. In recent years, interest in case-based reasoning has grown across the country. The two long-term agendas of case-based reasoning remain: Develop a scientific model of human memory, and build robust computer programs that can assimilate experiences and adapt to new situations. As the results of the past few years seem to demonstrate, these enterprises appear to be synergistic.

Acknowledgments

This work was supported in part by the Defense Advanced Research Projects Agency of the Department of Defense and the Office of Naval Research under contract N00014-85-K-0108 and the Air Force Office of Scientific Research under contract AFOSR-87-0295.

Bibliography

Alterman, R. 1986. An Adaptive Planner. In Proceedings of the Fifth National Conference on Artificial Intelligence, 65–95. Menlo Park, Calif.: American Association for Artificial Intelligence.

Ashley, K. 1987. Modelling Legal Argument: Reasoning with Cases and Hypotheticals. Ph.D diss., Univ. of Massachusetts.

Bain, W. 1986. Case-Based Reasoning: A Computer Model of Subjective Assessment. Ph.D. diss., Yale Univ.

Bain, W. 1984. Toward a Model of Subjective Interpretation, Technical Report, 324, Dept. of Computer Science, Yale Univ.

Bareiss, E. 1988. PROTOS: A Unified Approach to Concept Representation, Classification, and Learning. Ph.D. diss., Univ. of Texas.

Bareiss, E.; Porter, B.; and Wier, C. 1988. PROTOS: An Exemplar-Based Learning Apprentice. *International Journal of Man-Machine Studies* 29:549–561.

Bartlett, F. 1932. *Remembering*. Cambridge: Cambridge University Press.

Berger, J. 1989. ROENTGEN: A Case-Based Approach to Radiation Therapy Planning. In Proceedings of a Workshop on Case-Based Reasoning, 218–223. San Mateo, Calif.: Morgan Kaufmann.

Bower, G.; Black, J.; and Turner, T. 1979. Scripts in Memory for Text. *Cognitive Psychology* 11:177–220.

Bradtke, S., and Lehnert, W. 1988. Some Experiments with Case-Based Search. In Proceedings of the DARPA Workshop on Case-Based Reasoning, 80–93. San Mateo, Calif.: Morgan Kaufmann.

Branting, L. 1989. Integrating Generalizations with Exemplar-Based Reasoning. In Proceedings of a Workshop on Case-Based Reasoning, 224–229. San Mateo, Calif.: Morgan Kaufmann.

Buchanan, B.; Sutherland, G.; and Feigenbaum, E. 1969. HEURISTIC DENDRAL: A Program for Generating Explanatory Hypotheses in Organic Chemistry. In *Machine Intelligence* 4. Edinburgh: Edinburgh University Press.

Carbonell, J., and Veloso, M. 1988. Integrating Derivational Analogy into a General Problem-Solving Architecture. In Proceedings of the DARPA Workshop on Case-Based Reasoning, 104–124. San Mateo, Calif.: Morgan Kaufmann.

Collins, A., and Quillian, M. 1969. Retrieval Time from Semantic Memory. *Journal of Verbal Learning and Verbal Behavior* 8:240–247.

Cullingford, R. 1978. Script Application: Computer Understanding of Newspaper Stories. Ph.D. diss., Yale Univ.

DeJong, G. 1979. Skimming Stories in Real Time: An Experiment in Integrated Understanding. Ph.D. diss., Yale Univ.

Donahue, D. 1989. OGRE: Generic Reasoning from Experience. In Proceedings of a Workshop on Case-Based Reasoning, 248–252. San Mateo, Calif.: Morgan Kaufmann.

Hammond, K. 1989. *Case-Based Planning: Viewing Planning as a Memory Task*. New York: Academic.

Hammond, K. 1986. Case-Based Planning: An Integrated Theory of Planning, Learning, and Memory. Ph.D. diss., Yale Univ.

Hammond, K. 1984. Indexing and Causality: The Organization of Plans and Strategies in Memory, Technical Report, 351, Dept. of Computer Science, Yale Univ.

Hammond, K., and Hurwitz, N. 1988. Extracting Diagnostic Features from Explanations. In Proceedings of the DARPA Workshop on Case-Based Reasoning, 169–178. San Mateo, Calif.: Morgan Kaufmann.

Hayes-Roth, F.; Waterman, D.; and Lenat, D., eds. 1983. *Building Expert Systems*. Reading, Mass.: Addison-Wesley.

Hunter, L. 1989. Knowledge Acquisition Planning: Gaining Expertise through Experience. Ph.D. diss., Yale Univ.

Kass, A., and Leake, D. 1988. Case-Based Reasoning Applied to Constructing Explanations. In Proceedings of the DARPA Workshop on Case-Based Reasoning, 190–208. San Mateo, Calif.: Morgan Kaufmann.

Kintsch, W. 1972. Notes on the Structure of Semantic Memory. In *Organization of Memory,* eds. E. Tulving and W. Donaldson, 247–308. New York: Academic

Kolodner, J. 1988. Retrieving Events from a Case Memory: A Parallel Implementation. In Proceedings of the DARPA Workshop on Case-Based Reasoning, 233–249. San Mateo, Calif.: Morgan Kaufmann.

Kolodner, J. 1984. *Retrieval and Organizational Strategies in Conceptual Memory*. Hillsdale, N.J.: Lawrence Erlbaum.

Kolodner, J. 1980. Retrieval and Organizational Strategies in Conceptual Memory: A Computer Model. Ph.D. diss., Yale Univ.

Kolodner, J., and Cullingford, R. 1986. Towards a Memory Architecture That Supports Reminding. In Proceedings of the Eighth Annual Conference of the Cognitive Science Society. Amherst, Mass.: Cognitive Science Society.

Kolodner, J.; Simpson., R.; and Sycara-Cyranski, K. 1985. A Process Model of Case-Based Reasoning in Problem Solving. In Proceedings of the Ninth International Joint Conference on Artificial Intelligence, 284–290. Menlo Park, Calif.: International Joint Conferences on Artificial Intelligence.

Koton, P. 1989. SMARTPLAN: A Case-Based Resource Allocation and Scheduling System. In Proceedings of a Workshop on Case-Based Reasoning, 285–289. San Mateo, Calif.: Morgan Kaufmann.

Koton, P. 1988. Reasoning about Evidence in Causal Explanations. In Proceedings of the DARPA Workshop on Case-Based Reasoning, 260–270. San Mateo, Calif.: Morgan Kaufmann.

Lebowitz, M. 1980. Generalization and Memory in an Integrated Understanding System. Ph.D. diss., Yale Univ.

Mark, W. 1989. Case-Based Reasoning for Autoclave Management. In Proceedings of a Workshop on Case-Based Reasoning, 176–180. San Mateo, Calif.: Morgan Kaufmann.

Marks, M.; Hammond, K.; and Converse, T. 1988. Planning in an Open World: A Pluralistic Approach. In Proceedings of the DARPA Workshop on Case-Based Reasoning, 271–285. San Mateo, Calif.: Morgan Kaufmann.

Minsky, M. 1975. A Framework for Representing Knowledge. In *The Psychology of Computer Vision,* ed. P. Winston, 211–277. New York: McGraw-Hill.

Navinchandra, D. 1988. Case-Based Reasoning in CYCLOPS, A Design Problem Solver. In Proceedings of the DARPA Workshop on Case-Based Reasoning, 286–301. San Mateo, Calif.: Morgan Kaufmann..

Owens, C. 1988. Domain-Independent Prototype Cases for Planning. In Proceedings of the DARPA Workshop on Case-Based Reasoning, 302–311. San Mateo, Calif.: Morgan Kaufmann.

Quillian, M. 1968. Semantic Memory. In *Semantic Information Processing*, ed. M. Minsky, 227–353. Cambridge, Mass.: MIT Press.

Riesbeck, C. 1988. An Interface for Case-Based Knowledge Acquisition. In Proceedings of the DARPA Workshop on Case-Based Reasoning, 312–326. San Mateo, Calif.: Morgan Kaufmann.

Riesbeck, C., and Bain, W. 1987. A Methodology for Implementing Case-Based Reasoning Systems. Lockheed.

Riesbeck, C., and Martin, C. 1985. Direct Memory Access Parsing, Technical Report, 354, Dept. of Computer Science, Yale Univ.

Rissland, E., and Ashley, K. 1988. Credit Assignment and the Problem of Competing Factors in Case-Base Reasoning. In Proceedings of the DARPA Workshop on Case-Based Reasoning, 327–344. San Mateo, Calif.: Morgan Kaufmann.

Rumelhart, D., Lindsay, P., and Norman, D. 1972. A Process Model for Long-Term Memory. In *Organization of Memory*, eds. E. Tulving and W. Donaldson, 197–246. New York: Academic.

Schank, R. 1986. *Explanation Patterns: Understanding Mechanically and Creatively.* Hillsdale, N.J.: Lawrence Erlbaum.

Schank, R. 1982. *Dynamic Memory: A Theory of*

Learning in Computers and People. Cambridge: Cambridge University Press.

Schank, R. 1981. Failure-Driven Memory. *Cognition and Brain Theory* 4(1): 41–60.

Schank, R. 1980. Language and Memory. *Cognitive Science* 4(3): 243–284.

Schank, R. 1979. Reminding and Memory Organization: An Introduction to MOPS, Technical Report, 170, Dept. of Computer Science, Yale Univ.

Schank, R. 1975a. *Conceptual Information Processing*. Fundamental Studies in Computer Science, volume 3. Amsterdam: North-Holland.

Schank, R. 1975b. The Structure of Episodes in Memory. In *Representation and Understanding*, eds. D. G. Bobrow and A. Collins, 237–272. New York: Academic.

Schank, R. 1972. Conceptual Dependency: A Theory of Natural Language Understanding. *Cognitive Psychology* 3(4): 552–631.

Schank, R., and Abelson, R. 1977. *Scripts, Plans, Goals, and Understanding*. Hillsdale, N.J.: Lawrence Erlbaum.

Schank, R., and Abelson, R. 1975. Scripts, Plans, and Knowledge. In Proceedings of the Fourth International Joint Conference on Artificial Intelligence, 151–157. Menlo Park, Calif.: International Joint Conferences on Artificial Intelligence.

Schank, R., and Kolodner, J. 1979. Retrieving Information from an Episodic Memory, or Why Computers Memories Should Be More Like People's, Technical Report, 159, Dept. of Computer Science, Yale Univ.

Schank, R.; Collins, G.; and Hunter, L. 1986. Transcending Inductive Category Formation in Learning. *Behavioral and Brain Sciences* 9(4).

Schank, R.; Kolodner, J.; and DeJong, G. 1980. Conceptual Information Retrieval, Technical Report, 190, Dept. of Computer Science, Yale Univ.

Selfridge, M., and Cuthill, B. 1989. Retrieving Relevant Out-of-Context Cases: A Dynamic Memory Approach to Case-Based Reasoning. In Proceedings of a Workshop on Case-Based Reasoning, 313–317. San Mateo, Calif.: Morgan Kaufmann.

Shinn, H. 1988. Abstractional Analogy: A Model of Analogical Reasoning. In Proceedings of the DARPA Workshop on Case-Based Reasoning, 370–387. San Mateo, Calif.: Morgan Kaufmann.

Shortliffe, E. 1976. *Computer-Based Medical Consultations:* MYCIN. New York: American Elsevier.

Simpson, R. 1985. A Computer Model of Case-Based Reasoning in Problem Solving: An Investigation in the Domain of Dispute Mediation. Ph.D. diss., School of Information and Computer Science, Georgia Institute of Technology.

Skalak, D. 1989. Options for Controlling Mixed-Paradigm Systems. In Proceedings of a Workshop on Case-Based Reasoning, 318–323. San Mateo, Calif.: Morgan Kaufmann.

Stanfill, C. 1988. Learning to Read: A Memory-Based Model. In Proceedings of the DARPA Workshop on Case-Based Reasoning, 402–413. San Mateo, Calif.: Morgan Kaufmann.

Sussman, G. 1975. *A Computer Model of Skill Acquisition*. Artificial Intelligence Series, volume 1. New York: American Elsevier.

Sycara, K. 1988. Using Case-Based Reasoning for Plan Adaptation and Repair. In Proceedings of the DARPA Workshop on Case-Based Reasoning, 425–434. San Mateo, Calif.: Morgan Kaufmann.

Tulving, E. 1983. *Elements of Episodic Memory*. Oxford: Oxford University.

Tulving, E. 1972. Episodic and Semantic Memory. In *Organization of Memory,* eds. E. Tulving and W. Donaldson, 381–403. New York: Academic.

Turner, R. 1988. Organizing and Using Schematic Knowledge for Medical Diagnosis. In Proceedings of the DARPA Workshop on Case-Based Reasoning, 435–446. San Mateo, Calif.: Morgan Kaufmann.

Wall, R.; Donahue, D.; and Hill, S. 1988. The Use of Domain Semantics for Retrieval and Explanation in Case-Based Reasoning. In Proceedings of the DARPA Workshop on Case-Based Reasoning, 447–462. San Mateo, Calif.: Morgan Kaufmann.

Williams, R. 1988. Learning to Program by Examining and Modifying Cases. In Proceedings of the DARPA Workshop on Case-Based Reasoning, 463–474. San Mateo, Calif.: Morgan Kaufmann.

Woods, W. 1975. What's in a Link: Foundations for Semantic Networks. In *Representation and Understanding,* eds. D. Bobrow and A. Collins, 35–82. New York: Academic.

Zhang, X., and Waltz, D. 1989. Protein Structure Prediction Using Memory-Based Reasoning: A Case Study of Data Exploration. In Proceedings of a Workshop on Case-Based Reasoning, 351–355. San Mateo, Calif.: Morgan Kaufmann.

Stephen Slade is currently a lecturer in the Yale Computer Science Department, where he teaches AI. His interests include cognitive modeling, decision making, natural language processing, and political management information systems.