Cambridge University Press & Assessment
978-0-521-63302-4 — Dynamic Memory Revisited
Roger C. Schank
Excerpt
More Information

CHAPTER 1

Introduction to Dynamic Memory

What is a dynamic memory? It is a flexible, open-ended system. Compare the way an expert stores knowledge about books in his field to the way a library catalog system does the same job. In a library, an initial set of categories is chosen to describe a domain of knowledge. Within those categories, titles, authors, and subjects of the books are recorded. Such a system is not *dynamic.* Eventually, the categories will have to be changed; overutilized categories will require updating; other categories will have to be created to handle new subjects and subject divisions.

A library does not have a dynamic memory. It changes with great difficulty. More important, to change it requires outside intervention. An expert has neither of these problems. He can change his internal classification system easily when his interests change, or when his knowledge of a particular subject matter changes. For the most part, these changes are not conscious. The expert may relate one idea to another or he may fail to do so. He knows when he knows something, but there is a lot he doesn't know he knows. He may be able to categorize without knowing the categorization scheme he uses. He can make observations about what he knows and thus can alter the memory structures that catalog what he knows. He can do this without even realizing he has done it. He has a dynamic memory.[1]

Libraries require physical space, and decisions must be made about how to use it and when to leave certain areas open for future use. Knowing where you want to put a book, or information about this book, requires having some preconception of the possible places available for it in the library. But there would be disastrous consequences if our memories got stuck with "empty floors" awaiting collections that never materialized. Worse yet, imagine if we didn't have room in our category schemes for unanticipated new materials. We need a category

[1] For an example of research on category formation, see Ross, 1996a; Ross, 1996b.

scheme that can change not only as we acquire new knowledge, but also as we change our understandings (and thus classifications) of our knowledge.

The problem for libraries in this regard is no doubt great. People, on the other hand, seem to be able to cope with new information with ease. We can readily find a place to store new information in our memories, although we don't know where or what that location is. This is all handled unconsciously. We can also find old information, but again we don't know where we found it and we can't really say what the look-up procedure might have been. Our memories change dynamically in the way they store information by abstracting significant generalizations from our experiences and storing the exceptions to those generalizations. As we have more experiences, we alter our generalizations and categorizations of information to meet our current needs and to account for our new experiences (Wattenmaker, 1992; Ross, 1996b).[2] Despite constant changes in organization, we continue to be able to call up relevant memories without consciously considering where we have stored them. People are not aware of their own cataloging schemes, they are just capable of using them.

Consciousness does not extend to an awareness of how we encode or retrieve experiences. Our dynamic memories seem to organize themselves in such a way as to be able to adjust their initial encodings of the world to reflect growth and new understanding. Our memories are structured in a way that allows us to learn from our experiences. They can reorganize to reflect new generalizations – in a way, a kind of automatic categorization scheme – that can be used to process new experiences on the basis of old ones. In short, our memories dynamically adjust to reflect our experiences. A dynamic memory is one that can change its own organization when new experiences demand it. A dynamic memory is by nature a learning system.

Thinking about Artificial Intelligence

Prior to the writing of the first edition of this book, my primary focus in artificial intelligence (AI) had been on the problem of getting computers to be able to read. I had selected this focus because I felt that language and its use were a window into human understanding. After the first edition of this book was written, and in response to the

[2] Also Metcalfe (1993).

issues it raised, my colleagues and I began working on the issues of knowledge acquisition, learning, memory, and what came to be called case-based reasoning. Although we were successful in getting computers to read stories (see DeJong, 1977; Cullingford, 1978; Wilensky, 1978; Dyer and Lehnert, 1980 for descriptions of some of the programs we wrote at Yale), the creation of the programs themselves forced us to reconsider what reading a story means. Being able to answer questions about what was read, or summarizing or translating what was read, which were the tasks we chose for ourselves, was not sufficient to convince us or anyone else that these programs were actually reading.

The reason we were unconvinced was that these programs didn't remember, in any real sense, what they had read. Our programs were successful enough that we would show them off regularly. To do this, we had a ready supply of newspaper stories that we knew the computer would read accurately. I began to worry, however, that our programs never got bored. They read the same story about an earthquake in Iran over and over again, but never once exclaimed that there had been an extraordinary number of Iranian earthquakes lately or that they were mighty tired of reading the same story repeatedly.

It is hard to swallow the idea that a computer program that fails to remember what it has just read can be said to be comprehending. We tried, during the course of our work in creating computer programs that understand language, to avoid the issues of memory and learning. Language is hard enough to get a computer to process; did we have to work on memory and learning too? Well, yes, we did. The separation of language and memory is quite artificial. Linguists have always tried to separate the two to make their lives easier. Because I came from that tradition, it seemed reasonable to me to do the same. But the differences between people and computers began to nag me. People don't read yesterday's newspaper a second time unless they were powerfully impressed with it the first time or else have nothing else to read. People get bored and irritated by being asked to read something again and again, but our computer programs never did. They didn't learn from what they read. They merely coped with the mechanistic problems of language. That is, they were trying to piece out meanings without enhancing the meaning of those meanings to themselves. To put this another way, they didn't want to know the information contained in the stories they read.

Language is a memory-based process. It is a medium by which thoughts in one memory can, to some extent, be communicated in order to influence the contents of another memory. It is only one of several vehicles used to pass information from one memory to another. All of the senses can affect memory; language is an encoding of one kind of sense datum. Any theory of language must refer to a theory of memory, and any theory of memory is a theory of learning. No human memory is static; with each new input, with every experience, a memory must readjust itself. Learning means altering memory in response to experiences. It thus depends upon the alteration of knowledge structures that reside in memory.

In the work that preceded the original *Dynamic Memory*, we attempted to provide some view of how knowledge structures might represent information about events. In Schank and Abelson (1977), we developed the notion of a script. We defined script as a knowledge structure useful in the processing of text to the extent that it directed the inference process and tied together pieces of input. Input sentences were connected together by referring to the overall structure of the script to which they made reference. Thus, scripts were, in our view, a kind of high-level knowledge structure that could be called upon to supply background information during the understanding process. As embodied in the computer programs we wrote, they were essentially sets of predictions of event sequences. A script was constituted as a list of events that compose a stereotypical episode. Input events that matched one or more of the events in the list would cause the program to infer that the other events in the list had also taken place.

Since the publication of Schank and Abelson in 1977, some psychologists have found the notion of a script useful in explaining the behavior of children (Nelson, 1979; Nelson and Gruendel, 1979; McCartney and Nelson, 1981; Fivush, 1984; Slackman and Nelson, 1984; Adams and Worden, 1986; Hull Smith et al., 1989) and adults (Bower, 1978; Jebousek, 1978; Smith, Adams and Schorr, 1978; Graesser, Gordon and Sawyer, 1979; Graesser et al. 1980) engaged in the language comprehension process. Thus, some of the representations we proposed seem to have some psychological validity (Abelson, 1980; Graesser, 1981; Ratner, Smith and Dion, 1986; Bauer and Mandler, 1990; Farrar and Goodman 1992).

Considering language as a memory process changed our view of how understanding works. We now see language understanding as an integrated process. People don't understand things without making reference to what they already know and to what they think about what is being said. We don't break down the task of understanding language into small components. Rather, understanding is entirely a process of relating what we are hearing or experiencing to what we already know. In contrast to this, our early models were modular in nature, breaking the task of understanding into discrete and serially executed components. For example, SAM, our original story comprehension program, had a modular organization (Cullingford, 1978). After translating a sentence into Conceptual Dependency, the meaning representation scheme described in Schank (1972), SAM began the process of script application, which involves the recognition that a given script applies in a situation.

When a script was successfully identified, a set of predictions was made about what events were likely to transpire. We knew that such a modular approach was unrealistic – surely people begin to understand what a sentence is about before it is completely uttered. We built SAM in a modular fashion because that was the easiest way to work out the mechanics of script application; divorcing the parsing process from the process of script application simplified matters. But scripts are the sources of memories, and how we understand is affected by what is in our memories. A coherent theory of the structures in memory must naturally precede a complete theory of language understanding.

To solve problems involved in building understanding systems, we needed first to understand the kinds of high-level knowledge structures available in the understanding process. An important question that guides this work is, What else is available to an understanding system besides scripts? In the computer programs PAM (Wilensky, 1978) and POLITICS (Carbonell, 1979), we used plans and goals as high-level structures that control understanding. They served to help these programs make predictions based on story fragments. Certainly any memory must have access to such structures. But how many different sources of predictions are there? How can we find out what various knowledge structures are like?

The focus of the original version of this book was on building computer systems that understand. It became clear that to build such systems, we needed to build systems that learned as well. We know memories change over time. A person is changed in some way by every new sentence he processes. Smart computers would have to learn as well. However, we weren't (and aren't) interested primarily in computers. Computer programs help us make precise theories. They provide the rigor that is sorely lacking in most psychological theories. But, in the end, it is people who are the interesting subject. And, it became clear that if we wanted to know about how people processed language, we were going to have to study learning. So, we needed to study how the mind adapts to new information and derives new knowledge from that information. To do all this we needed a coherent theory of adaptable memory structures. We needed to understand how new information changes memory. We realized that learning requires a dynamic memory.

A dynamic memory would have to rely upon some scheme for structuring and restructuring its knowledge, a way of altering the structures it had previously found useful if their value faded or, alternatively, became more important. What is this system of organization, what do these structures look like, and how did they develop? It is unlikely that high-level structures in memory are innate. They develop because they address the needs that arise during processing by the individual understander, and different individuals have different needs. We know that our experiences affect the development of memory structures.

Scripts Revisited

In this book, I develop a theory of the high-level memory structures that constitute a dynamic memory. As mentioned, this theory of dynamic memory has its base in the older theory of processing natural language developed in Schank and Abelson (1977). From that earlier work came an idea critical to understanding how humans decide what to do and how they understand what others do – the concept of scripts described previously. In those days, scripts were intended to account for our ability to understand more than was being referred to explicitly in a sentence by explaining the organization of implicit knowledge of the world we inhabit. Thus, when John orders sushi, we assume that he is in a Japanese restaurant, that he might be seated at a sushi bar, and that he is probably using chopsticks and not a fork; we can even assume he is drinking Japanese beer. We assume these things because we know the sushi bar script. If we do not know this script, we cannot make such assumptions and thus might have difficulty understanding various sentences that refer to things we might be assumed to know.

Scripts enable people to understand sentences that are less than complete in what they refer to. When we hear "John ordered sushi but he didn't like it," we know that this sentence is referring to eating and to John's reaction to a type of taste sensation that he doesn't like. We know this because of what we know about restaurants (the restaurant script) and because of what we know about a small specification of the restaurant script, namely "sushi tasting." When we hear "John flew to New York, and he was very unhappy with the meal," we now must invoke the airplane script to understand the sentence. We do not imagine John flapped his arms to get to New York, nor that he was in a flying restaurant. We can explain what happened to him by saying that "airline food isn't very good," because we know the details of the airplane script, which include that kind of information.

We were originally interested in how a computer might process an experience such as this, and in that spirit we proposed that scripts and other knowledge structures were part of the apparatus that a knowledgeable entity would have to bring to bear during understanding. We endeavored to find out what kinds of structures might be available for use in processing. In general, we ignored the problems of the development of such structures, the ability of the proposed structures to change themselves, and the problems of retrieval and storage of information posed by those structures. We concentrated instead on issues of processing.

However, it seems clear in hindsight that no computer could simply be spoon-fed a script and be able to effectively function with it. What we know of restaurants is acquired in part by going out to eat, and by a process of constant reexamination of one's expectations in the light of their utility in processing a current experience. Thus, learning itself is at the heart of any knowledgeable entity because whatever knowledge might be attained by that entity would have to be malleable enough to be found and changed as a result of experience.

It is in this spirit that we must reexamine the notion of a script. Scripts have been taken to mean *some high-level knowledge source* and thus, given that there are probably a great many varieties of possible knowledge sources, different claims have been made for scripts that on occasion conflict with one another. In our early research we differentiated between plans and scripts, for example, but that distinction has not always been clear. Frequently, when we presented the issue of plans and goals, we were asked why *robbing a liquor store* was a plan and not a script, or why *reading the Michelin Guide* was not a script. The link between scripts and plans seemed fuzzy. For *liquor store robbing* to be a script and not a plan, it would have had to have been done a great many times. Of course such a thing might very well be a script for some people. We had chosen it as an instance of a plan because we were trying to illustrate the process of plan application, which was very different in nature from that of script application. In plan application, inferences about goals are made in order to establish the connections between input actions and the achievement of some goal. The most important point about script application is that often such goal-related inferences cannot be made. For example, without knowledge of the Japanese restaurant script, there is no way to determine why customers take off their shoes when they arrive.

The difference, then, between scripts and plans or any high-level knowledge structure resides in the amount of processing needed to come to understand a situation. The less one knows about a situation, or the less familiar one is with a certain kind of situation, the more inference work one has to do in order to process inputs dealing with that situation. Using scripts involves less work; planning implies more work. Scripts are a kind of mindless mental structure that allows one not to have to think too hard.

Our initial definition of a script was *a structure that describes an appropriate sequence of events in a particular context* or *a predetermined stereotyped sequence of actions that defines a well-known situation* (Schank and Abelson, 1975). The archetype that used scripts was SAM (Schank, 1975; Cullingford, 1978), a program to understand stories that used restaurants as their background. The idea was that we could short-cut the inference process by having certain inferences "come for free" because a script had been found to be relevant. When it was known that a story took place in a restaurant, all kinds of inferences – from table settings to check paying – became just that much easier to make.

Restaurant stories being neither plentiful nor very interesting, we began to look for new domains after we had initially demonstrated the power of script-based processing in our computer programs. We chose car accidents because of their ubiquity in the newspapers and their essential simplicity, and we began to alter SAM to handle these. Immediately we ran into the problem of what exactly a script was. Is there a *car accident* script? A computer could certainly use one to help it process such stories, but that would not imply that most people naturally would have acquired such a script. People who have never been in a car accident would not have a car accident script in the same sense that they might be said to have a restaurant script. Certainly the method of acquisition would be vastly different. Furthermore, the ordered, step-by-step nature of a script, that is, its essential stereotypical nature due to common cultural convention, was different.

This was emphasized in the way that a car accident script actually could be used to handle newspaper stories. Whenever a car accident occurred, we had to expect at least an ambulance script, a hospital emergency room script, a police report, possibly a subsequent trial script, and perhaps others as well, to be present. Were all these things really scripts? And, if they were, why was it that they seemed so different from the restaurant script in acquisition, use, and predictive power? To put it another way, it seemed all right to say that people know that in a restaurant you can either read a menu and order, or stand in line for your food. We felt justified in saying that there were many different *tracks* to a restaurant script, but that each of these tracks was essentially a form of the larger script. That is, they were like each other in important ways and might be expected to be stored with each other within the same overall outer structure in memory.

But what of accidents? Was there a general accident script of which collisions, accidental shootings, and falling out of windows were different tracks? Alternatively, was there a vehicle accident script of which those involving cars, trucks, and motorcycles were different tracks? Or was there a car accident script of which one car hitting an obstruction, two cars colliding, and chain reactions were different tracks?

It turned out for the purpose of creating SAM that none of this mattered. We encoded it all as scripts and allowed certain scripts to fire off other scripts to handle the sequence. However, the fact that we could make it work on a computer this way is basically irrelevant to the issue of the ultimate form and place of scripts in human memory. Did the fact that it worked in SAM really suggest that for people the emergency room script is in some important way a part of the car accident script? Although the idea of a general accident script seems to contradict an experientially based definition of scripts, that would have worked in SAM as well.

Gradually then, a practical definition of scripts was beginning to emerge that bore only surface similarity to the theoretical notion of scripts as a knowledge source for controlling inferences and tying together texts in highly constrained and stereotypical domains. This practical definition was that a script was a data structure that was a useful source of predictions. Scripts were supposed to depend on issues related to development based on repeated experience. But our use of scripts was not in agreement with our theory.

Whenever a script was accessed and an initial pattern match for an input made, the script could be used to predict what was coming next, or to take what did come next and place it within the overall pattern. In SAM's terms, a script was a gigantic pattern that could be matched partially in a piecemeal fashion.

This problem of precisely defining scripts became even more difficult when work began on FRUMP (DeJong, 1977). FRUMP is a program that was intended as a practical script-based approach to story understanding. SAM was rather slow and exceedingly fragile, since it made every inference within a script that was there to be made and, in doing so, had to rely upon an immense vocabulary and world knowledge store. Because the stories SAM read came from the newspapers, an unexpected vocabulary item was not only possible, it was rather likely. SAM had very little ability to recover from problems caused by missing vocabulary or missing world knowledge. (Later, a program was designed to take care of this to some extent; Granger, 1977.)

FRUMP got around these problems by relying more heavily on the predictive nature of scripts and less heavily on what the text actually said. FRUMP did not actually parse the input it received. Rather, it predicted what it would see and went about looking for words, phrases, or meanings that substantiated its predictions, relying upon what we termed *sketchy scripts* to do so. Examples of sketchy scripts included earthquakes, breaking diplomatic relations, wars, arson, and snowstorms. In other words, nearly anything at all could be considered a sketchy script (including robbing a liquor store, which put us back to square one). The theoretical difference between SAM's scripts and FRUMP's sketchy scripts is negligible. FRUMP's scripts are simply shorter and contain less information.

FRUMP's scripts are essentially just a set of "requests" (Riesbeck, 1975), which is another way of saying that they constitute a set of predictions about what might happen, and a set of rules about what to assume if those predictions are, or if they are not, fulfilled. But the concept of a script as an organized set of predictions is not exactly what we originally had in mind. It is easy to see why – for FRUMP – earthquakes and breaking diplomatic relations can be scripts.

But if earthquakes are scripts, then what is a script, anyway? Few of us have ever actually been in an earthquake; even fewer of us have, for example, broken diplomatic relations. But we do have knowledge about such events that can be used to understand stories about them or to handle similar situations. This knowledge can be encoded as sets