The Soar Cognitive Architecture and Human Working Memory

Richard M Young

Department of Psychology, University of Hertfordshire, Hatfield AL10 9AB, UK
R.M.Young@herts.ac.uk

Richard L Lewis

Department of Computer and Information Science, The Ohio State University, Columbus, Ohio
rick@cis.ohio-state.edu

Five central features of the theory

From the viewpoint of the Soar cognitive architecture, the term "working memory" (WM) refers to the psychological mechanisms that maintain information retrieved or created during the performance of a task. The following are the five key points made in the chapter concerning Soar's treatment of human WM:

(1) Soar is not specifically a "model of WM", but rather a cognitive architecture of broad scope, which focuses on the functional capabilities needed for a memory system to support performance in a range of cognitive tasks. The functions of working memory are distributed across multiple components of the architecture, including the long-term production memory.

(2) Even in a cognitive architecture with an unbounded dynamic memory, WM limitations can arise on functional grounds. Where such functional accounts exist, they take theoretical priority over capacity-based explanations of WM phenomena.

(3) Soar does not currently include any capacity limits on its dynamic memory (SDM), but is compatible with certain such limitations. In particular, a constraint that SDM can hold at most two items of the same "type" (suitably defined) yields a coherent explanation for many psycholinguistic phenomena in the comprehension of sentences. This constraint is motivated by computational efficiency concerns, and embodies the general principle of similarity-based interference (Baddeley & Logie; Cowan; Schneider; and O'Reilly, Braver & Cohen — all in this volume).

(4) Soar emphasises the role of learning in WM phenomena, even on tasks which experimentally are regarded as concerning just "performance." The moral is that WM cannot coherently be studied independently of long-term memory and while ignoring learning.

(5) Soar stresses the recognitional usage of information in long-term memory acquired as a by-product of earlier task performance. It therefore has close links to other approaches which emphasise the involvement of long-term memory in WM, and may be able to offer a computational process model for "long-term WM" (Ericsson & Delaney, this volume).

1. Introduction

In the Spring of 1987, Allen Newell delivered the William James lectures at Harvard University (Newell, 1990). He used the occasion to argue the case for striving towards theoretical unification in psychology, and to develop Soar, a new cognitive architecture, as the basis for a candidate unified theory. Soar had previously been presented as a problem-solving architecture for artificial intelligence (Laird, Rosenbloom & Newell, 1986; Laird, Newell & Rosenbloom, 1987). Since then, Soar, like the production system architectures from which it is derived (e.g. Newell & Simon, 1972), has been applied both within artificial intelligence as a vehicle for constructing knowledge-intensive systems, and within psychology for the modelling of human cognition. Although more work has been done on the artificial intelligence side (e.g. Tambe, Johnson, Jones, Koss, Laird, Rosenbloom & Schwamb, 1995), there has been a steady stream of work exploring how Soar offers state-of-the-art accounts of human empirical phenomena (Altmann, 1996; Altmann & John, in press; Howes & Young, 1996; Lewis, 1993, 1997a, 1997b; Miller & Laird, 1996; Polk & Newell, 1995; Wiesmeyer, 1992; Wiesmeyer & Laird, 1993).

Unlike smaller-scale theories in psychology, Soar is not shaped by the concerns of any single experimental area or paradigm. Soar therefore has no separate "memory mechanism", in the sense of components of the architecture aimed specifically at accounting for the results of experiments on human memory. Instead, the design of Soar is dominated by the functional considerations needed to exhibit general intelligence of human-like form (Newell, 1990). With regard to human memory, Soar's aim is not primarily to reproduce the findings of laboratory experiments, but to explain how memory works in the context of performing real, often complex, tasks. However, such an approach is useful even for the analysis of laboratory phenomena, because no psychological experiments test just memory. Instead, they all involve the entire cognitive system in the comprehension and performance of tasks.

The involvement of memory in complex cognitive tasks is, as Newell (1992) reminds us, an issue that memory theorists have largely neglected. In consequence, the "standard models" offered by most memory theorists are simply inadequate for explaining performance on cognitive tasks of any appreciable complexity. Decades of simulation work have shown that very small working memories — for example, those able to hold just a small number of items, say in the range 3 to 9 "chunks", or those holding 3-4 items in a "central executive" supplemented by phonological and similar loops — are simply incapable of supporting the performance of tasks such as problem solving or language processing (Broadbent, 1993). (For different reasons, similar doubts about the adequacy of the accounts of memory offered by mainstream cognitive psychology are expressed by other investigators, for example those concerned with studies of "practical memory" [e.g. Gruneberg, Morris & Sykes, 1988] or with the importance of "ecological validity" [e.g. Neisser, 1982, 1985].)

In this chapter we therefore examine the phenomena of human working memory (WM) from the viewpoint of a cognitive architecture of broad scope, which focuses on the functional capabilities needed for a memory system to support performance in a range of cognitive tasks. We argue for and demonstrate three points concerning the limitations of WM: (1) We show how the cognitive system, even with a limited-capacity short-term store, can handle complex tasks that require large quantities of information, by relying heavily on recognition-based long term memory working in concert with the external environment. (2) We argue that limitations on WM arise even in purely functional cognitive systems built without pre-set capacity constraints, and hence that empirically demonstrated limitations of effective WM do not necessarily imply a capacity-constrained underlying memory system. (3) We show how a specific mechanism of similarity-based interference can act as a resource constraint on the cognitive system, and offer a coherent account a wide range of psycholinguistic phenomena.

Terminology

Before we proceed any further, we need to deal with a few points of terminology that are otherwise guaranteed to cause confusion.

First, the Soar architecture includes an internal declarative memory that holds information specific to the task in hand, including control information. As with most production system architectures, that memory is known as the "working memory". Because that memory is not necessarily to be taken as the counterpart of human working memory, we will refer to it as *Soar's dynamic memory*, abbreviated SDM. We will reserve the term *working memory* (WM) for the psychological construct. We explain the relation between the two in the next section.

Second, as we shall see, a central role is played by Soar's learning mechanism through which it acquires new production rules. In Soar, that process of acquiring new rules is called "chunking", and the new rules themselves are called "chunks". Because the term "chunk" is already widespread in the literature on human memory with a different (though related) meaning, we will again avoid the term in its specialised Soar sense, and simply talk about Soar's *learning* or *acquisition mechanism*, and refer to the new productions as *acquired rules*.

These conventions should aid clarity within this chapter, but the reader who delves further will find that our usage is non-standard with respect to the Soar research literature[1].

Third, we use the terms "production", "rule", and even "production rule" interchangeably and without distinction.

Finally, we discuss Soar here in terms of its most recent version, Soar7. The technical concepts in Soar7 differ considerably from those of earlier versions. For example, where Newell (1990) talks of goals (which have states) and subgoals (which also have states), in this chapter we talk simply of states and substates. The change has no psychological significance, but again the reader should be aware that most of the Soar literature employs an older terminology.

2. Basic Mechanisms and Representations in Working Memory, and the Control and Regulation of Working Memory (Qs 1 & 2)

In order to identify those mechanisms in Soar that correspond to working memory, we must first provide a brief introduction to the Soar architecture. Next we will present a functional definition of working memory, abstracted from any particular mechanism that might realize it. We then describe how certain components of Soar realize the functions of working memory. The particular mapping we provide between Soar and human working memory is not the standard production system view, but we argue that it should hold for any *learning* production system.

---

[1] There is no primer or textbook for Soar, and Newell's (1990) book is now seriously out of date in several relevant respects. However, there is a user manual available (Congdon & Laird, 1995), and an on-line tutorial introduction to Soar for cognitive modelling (Ritter &Young, on line). There is also a "gentle introduction" to Soar (Lehman, Laird & Rosenbloom, 1996). Information about Soar can be found on the World Wide Web, and is most conveniently accessed through the list of Frequently Asked Questions: http://www.ccc.nottingham.ac.uk/pub/soar/nottingham/soar-faq.html .

Overview of basic Soar mechanisms

Soar is a cognitive architecture of broad scope, offered by Newell (1990) as a candidate "unified theory of cognition". Soar is best described at two levels, called the *problem space level* and the *symbol level*. At the problem space level, Soar casts all cognitive activity as transformations of *states* by *operators* within a state space. In a state space, there is a single *current state*, which encodes working information about the problem or situation being processed. Cognitive processing occurs by means of operators, which apply to the current state to yield a new current state. In straightforward cases, processing consists of a repeated cycle in which Soar first picks an operator, and then applies it to transform the current state into a new one.

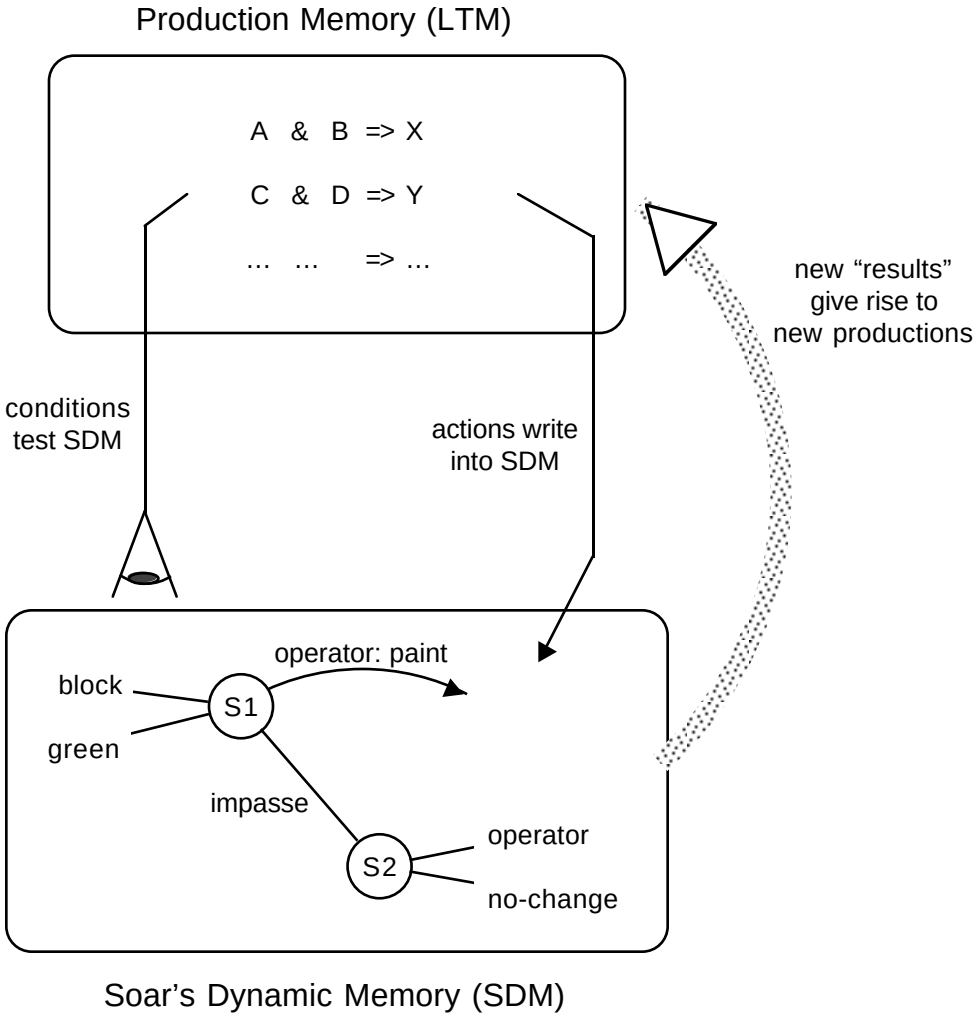

Figure 1. The main memories and processes in Soar.

This abstract description of Soar is realised concretely at the symbol level. At this level, the states and current operators are stored in *Soar's dynamic memory*, SDM, as shown in Figure 1. All persistent knowledge — including knowledge of when to propose operators, how to choose them, and how to apply them — is encoded in production rules. Each production represents a content-addressed piece of knowledge, which is sensitive to a particular pattern of information in SDM, and when it detects that pattern proposes particular changes or additions to SDM. All productions that are satisfied, i.e.

can find data in SDM to match their pattern, apply in parallel. Once all such rules have fired, their proposals are collected and the appropriate changes made to SDM. Usually, these changes will cause another set of productions to match, and the process continues.

At this level, Soar consists of two main memories: SDM, which declaratively holds dynamic information about the task at hand, and *production memory*, which holds persistent knowledge. Production memory is permanent. Once a rule is acquired, it is never lost (although there are ways in which it can be effectively masked or have its actions overruled).

Current versions of Soar employ a low-level, theoretically neutral representation of information in SDM, consisting of attribute-value pairs. Such a general-purpose representation can serve as a basis for building higher-level constructs of the kind familiar in cognitive science, such as schemata, frames, and so on. However, the representation is theoretically weak, and places little constraint on the form or content of information in SDM. Some work in Soar (e.g. Polk & Newell, 1995) has begun to explore the possibility of adopting a more constrained, theoretically stronger representation — specifically, that of "annotated models", inspired by the "mental models" of Johnson-Laird (1983). But the topic of representation remains an underdeveloped area for Soar.

Soar's "straightforward" processing described above, of repeatedly choosing an operator and applying it to the current state, can become blocked in various ways. For example, perhaps Soar knows of no operators to propose in the current state; or conversely, perhaps it proposes several but lacks the knowledge to choose between them; or perhaps it chooses an operator, but does not know how to apply it to the state. In these cases, Soar is said to encounter an *impasse*, where it is unable to proceed with its simple processing cycle. When this happens, Soar automatically sets up a new state (implicitly in a new state space), which is regarded as a substate to the state where the processing was blocked. The purpose of the substate is to trigger further processing (involving further implicit search of Soar's knowledge base) in order to retrieve information to resolve the impasse. For example, if the impasse arose because no operator was proposed, it can be resolved by finding a suitable operator. If the impasse arose because multiple operators were proposed, it can be resolved by finding knowledge to choose between them. And so on. Unsurprisingly, processing of the substate can itself encounter an impasse, which then leads to the setting up of a sub-substate, etc. So at any time, Soar may have a whole hierarchy, or "stack", of spaces it is working in, each with its own current state and (possibly) operator.

Soar's learning mechanism

Because all knowledge in Soar is encoded as production rules held in production memory, learning — in the sense of adding new knowledge — consists of acquiring new productions. The mechanism for doing this is intimately tied in with the impasses just described.

Whenever processing in a substate leads to a change being made to the superstate, that change is called a *result*, and Soar automatically acquires a new production rule. Recall that a production has a *condition* side and an *action* side. The action side of the new production is, obviously, the result, i.e. the change to the superstate. What should constitute appropriate conditions for the new production is less obvious. What Soar does, is to use for the conditions all those items in SDM which (a) are part of the superstate, and (b) are causally implicated in the sequence of production firings that led to the result. Here, an item being "causally implicated" means, roughly, that one of the firing productions matches against it.

Suppose that A and B (and perhaps many other items) are part of the current state in SDM when an impasse occurs. In the processing of the substate, suppose that a production matches against A and

adds X to the substate, another production matches against B and X and adds Y to the substate, and finally a third production matches against Y and adds the result R to the superstate. Then A and B, and only A and B, are both part of the superstate *and* matched against in the sequence of firings that leads to R, so they form the conditions, and Soar acquires a new production:

P1: A & B ⇒ R.

In future, whenever A and B are both on a state in SDM, the new production will fire, adding R to the state. If R was sufficient to resolve the impasse, then the impasse will henceforth be avoided.

Working memory in Soar

We have now described Soar completely enough to identify those mechanisms that correspond to working memory. But first, we must define clearly what we mean by *working memory*. Following the spirit of Miller, Galanter, & Pribram (1960), Baddeley & Hitch (1974), Just & Carpenter (1992) and others, we focus on the functional role of working memory in cognition. The following definition specifies that role independently of the mechanisms that realize it:

> *Working memory* refers to those computational mechanisms that maintain and provide access to information created or retrieved during the performance of a task. Any computationally complete system must support such functionality, because computation is inherently a process that requires the temporary storage and manipulation of partial and intermediate products.

It should be clear that Soar's dynamic memory (SDM) — its short-term declarative store — is precisely a component intended to serve the function of working memory. Indeed, that is why such components in productions systems are usually called "working memories." The traditional descriptions of Soar reinforce this identification by stating that the current "state" in the problem space is held in Soar's dynamic memory. Hence, it is natural to assume that the SDM directly corresponds to working memory.

However, making changes to SDM is not the only way that Soar can change state during the course of a task: the other way is for the content of the production rule memory itself to change, via Soar's learning mechanism. Because the rate of learning is a function of the rate of impasses, there are clearly some strong bounds on the rate at which the LTM can encode new state changes. Nevertheless, for all but the simplest immediate reaction tasks, there is enough time (on the order of seconds) for Soar to encode some intermediate products in LTM and later retrieve that information during the task. In effect, the LTM can become part of the functional working memory system.

We can illustrate the idea by thinking of the declarative dynamic memory and the long term production memory as being divided into four parts, according to the source and function of the constituent memory elements. First, there are those memory elements that were created prior to the task and are not used in the task (constituting a small fraction of SDM, but a large portion—nearly all—of LTM). Second, there are those elements that were created prior to the task and *are* used in the task (constituting a small part of SDM, and some small fraction of LTM). In LTM, this part constitutes the pre-existing skill for the task. Third, there are those elements that are both created and used during the task. This constitutes most of SDM, and again, a small fraction of LTM. *It is this third set of memory elements that plays the role of a working memory*. Finally, the fourth set consists of those elements created during the task, but not used.

This analysis leads to the inevitable conclusion that "working memory" in Soar consists of both SDM, in its familiar role, and also to some extent LTM, in its less familiar role of maintaining and retrieving intermediate products within a task. The components and processes of working memory,

therefore, include both the short-term dynamic memory, the long term memory, the matching of long term productions to the dynamic memory, and the learning mechanism that gives rise to new productions. We will see this involvement of LTM in working memory spelled out in more detail in the next few sections.

Relation to other theories of memory

The job of relating Soar's account of memory to others is spread throughout this chapter, but we make a start here with a first-cut comparison between what we have seen of Soar so far and other approaches.

We begin with the relatively easy topic of long-term memory. Most approaches to human memory impose a fairly clear-cut distinction between short-term and long-term memory (or memories). Soar is unequivocal in offering its production memory as the counterpart to human long-term memory. The various different kinds of aspects of long-term memory discussed in the literature — declarative, procedural, episodic, implicit, autobiographical — are not distinguished *structurally* in Soar. Differences among them would have to be reflected in the *content* of the knowledge in production memory (or perhaps, in some cases, in how that knowledge is used on different types of occasion). Soar's assumptions about the permanence of knowledge once acquired, and about its content-addressed, associative nature, place it near the mainstream of psychological theorising and obviously share in a long tradition. Soar's long-term memory is, though, marked out by its structural uniformity and the leanness of its assumptions.

With regard to the part of working memory realised by SDM, Soar shares an evident family resemblance with other production system cognitive architectures, such as ACT (Anderson, 1983, 1993; Lovett, Reder & Lebiere, this volume) and CAPS (Just & Carpenter, 1992). Again, however, Soar's relative austerity is notable: Soar has no apparatus for rule strengths or activation levels, and suchlike, and therefore has at least the potential for stronger theoretical accounts.

The comparison with models derived from experimental psychology is less straightforward. The sharp two-way division between permanent and ephemeral memories has obvious echoes of earlier ideas about "short term" and "long term" memory (e.g. Murdock, 1963) or "primary" and "secondary" memories (Waugh & Norman, 1965; Atkinson & Shiffrin, 1968) — and consequently, in the eyes of some memory psychologists, gives Soar an old-fashioned look. The most historically influential touchstones are probably the idea of a short-term memory for the temporary storage of information (e.g. Miller, 1956), and its later elaboration into a "working" memory for transient, task-relevant information, particularly under pressure of data from study of the immediate recall task and its variants (Baddeley & Hitch, 1974; Baddeley, 1986; Baddeley & Logie, this volume). In relation to both of these, we need to repeat warnings we have already given in this chapter: Complex cognitive tasks simply *cannot* be performed with temporary storage of only 7 ± 2 items; Soar's SDM does not have an imposed capacity limit; and SDM should not be simplistically identified with the psychological notion of WM. To some extent, Soar can be seen as complementary to the main thrust of the last decade of WM work, since it focuses primarily on the so-called "central executive" concerning which WM researchers have, at least until recently (Baddeley, 1996), been so reticent.

3. Learning, Knowledge and LTM (Q 6)

The next two sections explore some of the surprisingly far-reaching implications of Soar's learning mechanism for human memory and learning. We first consider the unique theoretical role that

learning has in cognitive models built in Soar, and then look at some examples of the mechanism in action. These examples lay the foundation for understanding precisely the role of Soar's learning and LTM in realizing working memory functions.

Where do the productions come from?

We start this section with a brief look at a question often asked about a Soar model, namely, where do the productions come from?

One answer is, simply enough, that they are postulated by the theorist. In this, Soar differs not at all from any other kind of model. However, Soar offers also a second, less familiar, answer to the question of the origin of the production rules, which is that it learns them — or at least, some of them — for itself. An ideal cognitive model would learn all the relevant productions itself. Of course, in practice that is not possible. For a particular model, some hand-written rules have to be provided by the theorist as a starting point (although it is now quite common in Soar models for learned rules greatly to outnumber the hand-written ones, e.g. Altmann, 1996; Altmann & John, in press). The aim then is to provide, initially, only capabilities which it is reasonable to suppose a human subject has when he or she walks into the experimental situation. If a Soar model is able to interpret simple English sentences (Lewis, 1993), uses that ability to understand instructions (Huffman, 1994; Huffman & Laird, 1995), and by so doing acquires productions for performing a cognitive task (Lewis, Newell & Polk, 1989), then those resulting productions have a stronger theoretical status than if they were simply written by hand, i.e. "programmed" by the theorist. (Newell, 1990, refers to this methodology as one of "reducing the theorist's degrees of freedom".)

Rosemary's baby

Consider the following situation. You know that Rosemary, a friend of yours married to Robert, has just had a baby, but you do not know whether it was a boy or a girl. You want to find out which it is, so that you can send Rosemary a congratulation card, buy a present, and generally be in a position to say the right things to Rosemary when you see her. You obviously cannot determine the sex of the baby by sitting down and *thinking* about it, so on the next suitable occasion you ask Robert, and he tells you that their baby is a girl. To learn this information means to remember it, so that in the future you will know the sex of Rosemary's baby without having to ask again. How can we model the cognitive (if not the social) aspects of this scenario in Soar?

The net effect of the learning process should be that whenever the Soar model wonders about the sex of Rosemary's baby, it retrieves the information that the baby is a girl. This means that the model needs, somehow, to acquire a production we can write as:

```
P2:   what-is (sex-of, Rosemary's-baby)  ⇒  girl.
```

An acquired rule of this form is described as a *recall* rule, because it serves to recall, i.e. into SDM, an item of information which is known, i.e. stored in production memory.

The question then is, how does the Soar model acquire that rule? One might imagine the following kind of route. In some state, Soar wants to know the sex of Rosemary's baby. It doesn't *know* the answer, i.e. it doesn't yet have a production like P2, so there is an impasse and a substate is set up. The processing on the substate, in order to resolve the impasse, decides to go and ask Robert, and that subtask temporarily takes precedence in the main state over the primary task of finding the baby's sex. So Soar wanders off … and a little while later Robert tells Soar "It's a girl". With that

information in hand, the substate can finish the task by marking 'girl' on the superstate. That result, from a substate to a superstate, causes Soar to acquire a new production.

The trouble is, that a Soar model operating along those lines acquires, not P2, but the somewhat different rule:

```
P3:   what-is (sex-of, Rosemary's-baby)  &  answer=girl  ⇒  girl.
```

P3 is the same as P2, except that it has the extra condition "answer=girl". That condition is included because in the processing that led to the result, some production must have matched against Robert's answer "It's a girl" in order for the right result to be found. The item "answer=girl" is therefore causally implicated in the result, and consequently included in the acquired rule.

The extra condition prevents P3 from doing the job we want. A production like P3 is known as a *circular recognition rule*. It cannot recall information, but it can recognise the answer in the sense that it fires when the right information is in SDM. But that still leaves us with the problem of explaining how the recall rule, P2, is acquired.

A multi-stage learning model

To explain how P2 arises, we temporarily ignore the question of learning and focus instead on the task of obtaining the answer to the question "what-is (sex-of, Rosemary's-baby)". A Soar model has available to it four potential routes to the answer:

1. If Soar already has a rule like P2 which recalls the information, then it fires and the task is done. There will be no impasse.
2. If Soar happens to have the information already in SDM, then it is picked up and returned as the answer to the question.
3. Soar can generate the possible answers to the question (namely, "boy" and "girl"), and see if either of them triggers a recognition rule like P3. If so, that item is returned as the answer to the question.
4. Soar can seek the answer from an external source, such as by telephoning Robert and asking him.

These four routes are listed in order of increasing mental and (in the case of 4, physical) effort. Accordingly, it is rational for a Soar model to try the different routes in that order, with the easy ones first. The behaviour of such a model, over a sequence of trials, is instructive. We assume initially that neither P2 nor P3 is present:

- On the first trial, neither route 1 nor route 2 is applicable. Soar tries route 3, but fails because there is no recognition rule. It therefore adopts route 4, asks Robert for the information, and is told "It's a girl". With that information in SDM, route 2 can apply, so Soar picks up the reply and returns the answer "girl". We have already seen that the consequence of that behaviour is that Soar acquires the recognition rule, P3.
- On the second trial, again neither route 1 nor route 2 is applicable. Soar tries route 3, and asks itself in turn "boy?", "girl?". For the "girl" option, rule P3 fires, indicating that it recognises "girl" as the answer it has previously seen. Soar can therefore return "girl" as the result. In so doing, it acquires a new rule, P2. This time, the unwanted extra condition is not included in the rule, because the Soar model has generated the answer itself (in a substate) instead of having to rely on its already being present in SDM. The "answer=girl" is therefore not this time causally implicated in the result, and does not appear in the rule conditions.

- On the third trial, and subsequently, recall rule P2 fires, implementing route 1. The Soar model has therefore now "learned" the sex of Rosemary's baby.

A couple of features of this story about learning are worthy of comment. First, the model engages in no deliberate activity of "learning". Instead, the learning occurs as a side-effect of doing the task, as a consequence of Soar's automatic mechanism for acquiring new rules. Second, this kind of learning is inherently multi-pass. The first trial is needed to acquire the recognition rule. The second trial makes use of the recognition rule to pick out the right answer from among those generated, and leads to the acquisition of the recall rule. Only from the third trial onwards does the recall rule itself come into play. Third, the learning relies upon a process of *generate and recognise*, in which the learner generates possible options and makes use of previously acquired knowledge to recognise which is right. This process has evident similarities to other generate-and-recognise models previously proposed for memory (e.g. Bahrick, 1970; Kintsch, 1970; Anderson & Bower, 1972).

Implications for long-term memory

Although the model just presented is for a simple and somewhat contrived task, the approach and its characteristics generalise to a wide range of learning situations, basically those where crucial information has to be acquired in order for the task to be performed. (*Post-event learning*, where the learner has to remember the *outcome* of an action or event, is a little different in detail, but it shares the same generate-and-recognise basis.) Much of the work in Soar has dealt with the learning of instructions (Huffman, 1994; Huffman & Laird, 1995; Howes & Young, 1996), where the models exhibit a gradual progression from a performance that is initially externally based, i.e. dependent upon interpreting the externally-presented instructions, through a stage of recognition-based learning, to a fully internalised and assimilated skill.

Vera, Lewis & Lerch (1993) add an important ingredient to the story, by analysing the kind of situations which occur often in practice, where the alternative answers are provided by the external environment and therefore do not have to be generated from the learner's internal, cognitive resources. In learning to use a bank automated teller machine (ATM), for example, the action at each step makes use of a specific external object: to press one of the labelled buttons, or type on the numeric keys, or insert a card into the card slot, and so on. To acquire the correct sequence, the learner needs to supply only the recognitional knowledge. As further discussed by Vera *et al* (1993) and by Howes & Young (1996), Soar predicts that the skill acquired by the use of such external generators is strongly "situated" or "display based" (Larkin, 1989), in the sense that a person can perform the task fluently when in the task environment, but cannot produce the learned procedure — for example, by describing it — away from the situation. This inability is known to be characteristic of expert skill (Mayes, Draper, McGregor & Oatley, 1988; Payne, 1991), and has been regarded as puzzling.

Howes & Young (1996, in press) review these findings and describe ways in which Soar's learning mechanism serves to constrain the kinds of models that can be acquired, thereby making architecturally-based predictions about the corresponding human cognitive skill. In the area of learning to use interactive computer systems, the main differences in learnability between interfaces stem from the relative ease or difficulty of the generate part of the process. In addition to the display-based nature of the learned skill, Howes & Young show that the learning mechanism predicts that, given certain common properties of the device, a display-based (e.g. menu) interface is necessarily easier to learn than a keyboard-based (e.g. command line) one; and that for keyboard-based devices, learning is necessarily easier with either meaningful (rather than meaningless) or mnemonic (rather than arbitrary) command names.

In this section we have examined some of the implications of Soar's learning mechanism for long-term memory, learning, and cognitive skill. The next section focuses more specifically on the consequences of that mechanism for WM.

4. Complex Cognitive Activities (Q 5)

Because Soar is constructed with the aim of providing the functional capabilities for an intelligent agent, it is on its home ground when being used to model memory and learning in complex cognitive activities. In this section we briefly describe three studies bearing on the role of WM in complex tasks. All three studies model situations where a human subject has to deal with large quantities of information relevant to the external situation. In all three cases, trying to hold that information in SDM is not a functionally adequate solution. Instead, these models make use of Soar's learning mechanism to hold the information in LTM, and perform the task with a bounded SDM. The picture that emerges from these studies is of how Soar's learning mechanism, and the strongly recognition-based processing it implies, provide an adequate basis for complex cognitive processing within a limited WM. These results are related to Ericsson & Kintsch's (1995) ideas about "long-term working memory".

Howes' model of recognition-based search

Howes (1993, 1994) describes a model, called Ayn, capable of searching, and learning to navigate its way, through large external spaces. It deals with an environment characterised as a set of *locations* connected by *paths*. It can deal with tasks such as finding its way through a hierarchical computer menu system, or through the streets in an unfamiliar town centre. In the former case, the locations are the possible menus and the paths are the choices from each menu. In the latter case, the locations are street intersections and the paths are the streets leading from each intersection.

Ayn works by exploiting the production rules acquired as a by-product of the process of dealing with locations and considering the choice of possible paths, and using them as the basis for a (long-term) recognitional memory. Each time Ayn encounters a location, it attempts to comprehend it, to see what clues (if any) it offers to the correct route. And each time Ayn considers a path, it sees if it has any basis for evaluating that move. Those processing steps lead to productions being acquired which, as with the example of Rosemary's baby in the previous section, can serve to recognise which locations or paths Ayn has seen before, and which it has not. Ayn uses various heuristics that exploit the recognitional knowledge to improve its search. For example, if Ayn recognises its goal as one that it has reached before, then it knows the goal can be reached by sticking to paths it has taken before, so it prefers paths it recognises to those it does not. Over repeated trials, Ayn gradually improves its knowledge of the network to the point where it can navigate to familiar targets without error.

Because Ayn relies upon recognitional knowledge, which is stored in Soar's production memory (corresponding to human LTM), it avoids making heavy demands upon SDM, and Ayn in fact will search a space of any size using an SDM of fixed capacity. Ayn minimises the deliberate processing that has to been done in order to learn the network. Instead, as with the example of Rosemary's baby, Ayn makes recognitional use of knowledge acquired as a side-effect of performing the main task.

Translated back into terms of human cognition, the Ayn model demonstrates how, even with a strictly limited short-term store, people can search an external space of indefinite size, by relying upon recognitional use of knowledge acquired in LTM.

Altmann's model of episodic memory for external information

Altmann (1996; Altmann, Larkin & John, 1995; Altmann & John, in press) analysed and modelled the behaviour of a skilled programmer working at a task of understanding and modifying a large computer program. She works primarily with a single window on a computer screen which, as she issues commands and requests new information at the bottom of the window, slides the older information up and eventually off the screen. At various points in her investigation, the programmer encounters items about which she knows she has previously seen some information, and she then retrieves that information for further processing by scrolling back through the display window. The interesting feature of this task is that the programmer is being presented with far more information and detail on the screen than she can possibly retain. Yet, she manages to remember enough about the existence and location of relevant external information that she can find it again when she needs to, in order to dig deeper.

Altmann's model attempts to comprehend each new item of information that appears on the screen, although this comprehension can vary enormously in its depth and completeness. Each act of comprehension results in the acquisition of one or more new production rules, which include the salient features of the item being processed and a (very crude) indication of the circumstances in which the item appears. Such rules form effectively a (very simple) episodic trace of what has been seen. Even though only a tiny fraction of the content of the computer display is being captured in this way, these productions arising as a side-effect of the comprehension can serve as recognitional knowledge. When the model encounters an item about which more information is needed, if one of these recognition rules fires then Soar knows that it has seen the item before; and if the remembered context indicator differs from the current one, then Soar knows it must have seen the item earlier, and so can find it by scrolling back.

The implications of this model for human memory both reinforce and extend those we have seen from the previous examples. Once again, we have the idea of learning occurring as a side-effect of processing, with heavily recognitional use being made of the production rules acquired as a by-product of that processing. In this case, the model shows how a human programmer can, within the constraints of a limited WM, make effective use of recognitional knowledge in LTM to index into and navigate around a large and growing display of external information.

Altmann and colleagues relate their results to the idea of long-term working memory (LT-WM) propounded by Ericsson & Kintsch (1995; Ericsson & Delaney, this volume). Altmann's model shares with Ericsson & Kintsch's analysis the core suggestion that LTM be used to provide access to a larger set of information than can be carried in WM. Soar adds to LT-WM the twist that the relevant knowledge in LTM is primarily recognitional in character, and serves to provide access to a potentially vast body of external information. Unlike the cases studied by Ericsson & Kintsch, Altmann's programmer does not put effort into the deliberate construction of retrieval cues for the information being memorised. Altmann *et al* ascribe this difference to the contrasting requirements of the tasks. Altmann's programmer encounters a huge amount of information, only a small part of which will ever be sought again, so there is no possibility of building specific retrieval structures. Instead, Altmann *et al* propose extending LT-WM with the idea of a ubiquitous "episodic LT-WM", which the cognitive architecture acquires passively and pervasively.

IDXL model of exploratory search

Rieman, Young & Howes (1996) describe a Soar model called IDXL, which performs the kind of exploratory search that experienced computer users engage in when asked to use an unfamiliar menu-driven application. IDXL accounts for many of the empirical phenomena observed with this kind of exploratory learning (Rieman, 1994, 1996; Franzke, 1995). Users typically cannot and do not go directly to the right menu item. Instead, they examine the menu headers, pull down and browse through some of the menus, focus on a few promising items, and may end up visiting the correct item several times, often for increasing durations, before finally choosing it.

IDXL pays attention to one item on the screen at a time, performing some processing on an item and then moving to an adjacent one. The processing of each item is done locally, with information being held in SDM about only the individual item being attended to. Each time an item is examined, a quantum of processing is performed to assess its relevance to the task being done, and leads to a production being acquired to summarise the outcome of that assessment. Over multiple visits to an item, an increasingly detailed and reliable evaluation of the item is represented in the acquired rules. Those rules are also used recognitionally, for IDXL to know on each visit to an item what processing it has already received.

IDXL's implications for human memory agree substantially with those of the previous examples. IDXL shows how experienced users can explore a potentially large, externally presented space of possible actions with only a fixed-capacity WM, by storing information gathered during the exploration in LTM, and using it recognitionally.

5. Limitations of WM: Functional limitations (Q 4)

In the previous section, we discussed three Soar models that perform complex cognitive tasks while operating within a limited working memory. That is only half the story (though a crucial half): we must now demonstrate how WM limitations might actually arise in Soar. In this section and the next, we present additional models that illustrate two theoretical points about WM limitations in Soar. First, there are several ways in Soar that phenomena attributed to limited WM may arise from the interaction of functionally motivated aspects of the architecture, without assuming any capacity constraints. Second, independent efficiency concerns in Soar lead us to posit a simple but severe representational constraint on SDM that is consistent with a well-known principle of human WM limitations, similarity-based interference. This constraint leads to detailed accounts of a number of psycholinguistic phenomena in sentence processing, while still allowing the complex task of comprehension to proceed.

Functional limitations and resource limitations

The study of working memory is dominated by the topic of limitations. The majority of experimental studies demonstrate that people can hold a certain amount of information in WM, but no more, under such-and-such conditions. The most famous case, deriving from Miller's (1956) celebrated paper, is that of limited immediate memory capacity. Usually these limitations are seen as being due to underlying resource or implementational constraints. For example, the decay rate of the phonological loop (Baddeley & Logie, this volume) is a hard constraint that presumably reflects properties of the neurobiological substrate. The same is true for models positing activation limits (Lovett, Reder & Lebiere, this volume), or limits of attentional span (Cowan, this volume). The implicit assumption is that if the hardware of the brain had different properties—were somehow better

designed—then we would be able to remember more, understand more complex language, think and learn faster, etc.

The idea of resource constraints appears not to sit comfortably with Soar. It has no mechanisms of quantitative gradation, such as *activation level* or *trace strength*[2], and attempts to impose a brute force limitation on the size of SDM — to decide that it cannot hold more than seven items, say, or seventeen — seem not to work well. This aspect makes many psychologists doubtful about Soar (e.g., the commentaries in Newell, 1992), but as we shall see it does not rule out the possibility that Soar can model and account for the psychological phenomena associated with limited WM.

Newell (1990) points to another possible source of performance limitations. He distinguishes between resource constraints and *functional limitations*, which can arise from the interaction of cognitive components motivated on purely functional grounds. In other words, they derive from considerations of the task an agent is trying to perform and what it has to do in order to perform it, without needing to appeal to properties of the implementational substrate. The existence of functional explanations of performance limitations is theoretically very important. If an explanation of a phenomenon can be offered on independently (and functionally) motivated grounds, then no further explanation need be sought in terms of implementational or resource constraints. In the following subsections we consider two examples of Soar models that exhibit such functional limitations.

Example 1: Performance limitations and individual differences in syllogistic reasoning

It has long been known that certain forms of syllogistic reasoning are difficult for most people to perform. For example, the syllogism in (1) below is easy to solve correctly:

(1) All artists are beekeepers.

All beekeepers are chemists.

What (if anything) necessarily follows?

It necessarily (and quite transparently) follows from the two premises that *all artists are chemists*. However, syllogisms such as (2) below are notoriously difficult to get right:

(2) No artists are beekeepers.

Some beekeepers are chemists.

What (if anything) necessarily follows?

The correct response to (2) is *some chemists are not artists*, though most subjects respond with *no artists are chemists*, *some artists are not chemists*, or *no chemists are artists*.

Why are certain forms of reasoning so difficult? Johnson-Laird (1983, 1989) has championed an approach to human reasoning which claims people (who are not trained in logic) approach such

---

[2] However, that does not mean that Soar cannot account for certain graded phenomena. See, e.g., Miller & Laird (1996).

puzzles, and all kinds of everyday reasoning, by building and manipulating mental models of the situations described. A model is a representation containing elements that correspond directly to the individuals or objects in a situation. Models are to be contrasted with logic representations, which represent via a set of logical sentences or axioms, and which are manipulated via proof procedures. The evidence Johnson-Laird garners in favor of the mental model theory is that reasoning difficulty is primarily a function of the number of models that must be constructed to solve a problem, not a function of the complexity of the logical proof (Johnson-Laird, 1988). The relation to working memory is that some reasoning tasks are difficult because they require the construction and maintenance of multiple models, which taxes a limited capacity working memory.

Polk & Newell (1995) offer an alternative theory that also assumes people construct mental models in verbal reasoning tasks, but does not assume that they construct multiple models. Polk & Newell argue that people solve verbal reasoning tasks, such as the three-term syllogisms exemplified above, by using their pre-existing language skills. As a weak claim about a component process of reasoning, this is uncontroversial and undeniably true. But the theory makes a much stronger claim: verbal reasoning is accomplished *exclusively* by existing comprehension and production skills, in particular, by comprehending and recomprehending the premises, i.e., building an initial single model and incremental augmenting that model. There are no special reasoning processes—not only are there no logical proof procedures, but there are also no special processes to support logical reasoning by constructing and manipulating multiple models.

The theory offered by Polk & Newell (1995) has a number of properties relevant to the question of functional constraints. First, the model is strongly functional. One sense of this claim is simply that the model performs the task, i.e. it reads the two premises and attempts to draw a conclusion. The sense more directly relevant here is that the model is functional in that it draws on abilities, like language skills, that exist in order to provide the agent with certain functional capabilities (such as speaking, listening, reading) motivated independently of this particular task.

Second, subjects' difficulties with the task are explained as a consequence of this functional basis. Polk & Newell point out that although subjects' everyday language skills (e.g. of encoding and comprehension) can serve to perform the syllogism task, they are not perfectly adapted to that task, and so by themselves are not adequate for perfect performance. For example, the model resulting from language comprehension "is guaranteed to represent a situation in which that proposition is true. But, in general, … an annotated model may encode information that is not inherent in a proposition (and be unwarranted) or fail to encode information that is (and be incomplete)." (Polk & Newell, 1995). In this and similar ways, difficulties with the task are explained on the grounds of functional constraints without appeal to the notion of resource constraints.

Third, the model deals with individual differences. The model contains a number of qualitative parameters that determine such things as detailed aspects of how the premises are encoded, for example, whether or not a premise of the form *No X are Y* will be interpreted as also implying *No Y are X*. Different choices for the detailed structure generate a whole space of models, and Polk shows that different models in that space provide a good fit to different individuals. The fit to individual subjects is in fact at least as good as the subjects' test-retest fit to their own performance one week later. This parameterised model provides a highly successful account of individual differences, which makes no reference to resource constraints.

Example 2: Performance limitations and individual differences in ambiguity resolution

We now turn to another aspect of verbal processing in which capacity limitations are implicated: ambiguity resolution in sentence comprehension. There is a long-standing and continuing controversy in psycholinguistics over the extent to which non-syntactic factors can affect the initial structure assigned in parsing. Consider sentences (3) and (4), which contain a momentary ambiguity that is resolved later in the sentence:

(3) a. The defendant examined the courtroom.

b. The defendant examined by the jury was upset.

(4) The evidence examined by the jury was suspicious.

The verb *examined* is ambiguous: it may be interpreted as the main verb (as in 3a), or as starting a relative clause (as in 3b and 4). Researchers have argued for a variety of factors that determine initial ambiguity resolution, ranging from purely structural properties (e.g, Ferriera & Clifton, 1986) to lexical semantics, lexical and structural frequencies, or referential properties (e.g; MacDonald, Pearlmutter, & Seidenberg, 1994). For example, the inanimacy of the subject (*evidence*) in (4) provides a cue that the verb *examined* should be interpreted as a relative clause, because that is the most plausible interpretation (inanimate objects usually do not examine things). Indeed, Trueswell et al (1994) and others have found that such semantic constraints can have a strong effect on ambiguity resolution.

Just & Carpenter (1992) added a novel twist to the debate, by demonstrating that ambiguity resolution behavior is a function of individual differences as well as linguistic factors. In particular, they argued that the behavioral differences are due to differences in working memory capacity. People with high working memory capacity (high-span) are able to appeal to nonsyntactic factors, such as the animacy features in (4), while people with low working memory capacity (low-span) do not have the capacity to be sensitive to such factors. They found that low-span subjects performed in a modular fashion on material such as (4), while high-span subjects performed in an interactive fashion on identical material. Thus, they argue, modularity is not an all-or-none architectural issue, but a graded function of working memory limitations.

Lewis (1993) presents an alternative explanation of these individual differences that does not appeal to limited working memory capacity to account for the modular behavior of the low-span subjects. The explanation is derived from NL-Soar, a Soar model of language comprehension. In NL-Soar, local ambiguity manifests itself by the simultaneous proposal of multiple operators corresponding to the different interpretations. For example, in (3) and (4) above, at the verb *examined*, two operators are proposed: one corresponding to the main verb structure, the other corresponding to the relative clause structure. Ambiguity resolution then takes place by drawing on available search control knowledge. In the case of the ambiguity in (3) and (4), a search control production can test the proposed operators and the semantic content of the subject (e.g, *evidence*), and prefer the relative-clause operator in the appropriate contexts. Thus, NL-Soar can model the rapid, on-line effects of semantic context observed in the interactive studies (e.g., Trueswell *et al*, 1994) and in the high-span subjects of Just & Carpenter (1992).

However, nothing guarantees that such a search control production will be available. If the knowledge is present only through deliberate processing in sub-states, there may not be enough time to perform all the inferences necessary to make the right selection. Under press of time, there may be no alternative but to select one interpretation by some default preference. In such a case, NL-Soar behaves in a modular fashion since the required knowledge sources are not applied on-line.

Thus, like the explanation of individual differences in verbal reasoning discussed above, the NL-Soar theory attributes the individual differences in ambiguity resolution behavior to differences in parsing skill. High span subjects are not necessarily performing well because of their higher working memory capacity, but because they have been exposed to the appropriate linguistic material which led to the learning of specific, semantically-sensitive ambiguity resolution rules. (Lewis, 1993, describes how NL-Soar is in fact capable of learning such rules).

Related approaches

The focus on knowledge differences as an explanation of individual differences is shared by the long-term working memory approach of Ericsson & Delaney (this volume) (see also Ericsson & Kintsch, 1995, which suggests a similar account of the Just & Carpenter data). Furthermore, the methodological approach of positing cognitive resource limitations only as a last resort is a prominent feature of the EPIC research (Kieras & Meyer, this volume). The point is that *some* (though not necessarily all) performance limitations may be due to knowledge or skill differences and purely functional aspects of cognitive architecture. The novel aspects that the Soar theory adds include an account of how the relevant skill differences can actually arise as a function of experience, and a precise explanation of how functional aspects of the architecture can lead to performance limitations.

6. Limitations of WM: Resource constraints (Q4)

The previous sections demonstrated how performance limitations may arise in a functional model without assuming any resource or capacity constraints. Of course, this does not deny that there are some kinds of hard resource constraints. We now consider one way of incorporating such constraints into Soar, and apply the resulting theory to the domain of syntactic processing, again within the NL-Soar model discussed above. The constraint is motivated by efficiency concerns in Soar, and yields a simple theory of capacity limitations that accounts for a wide range of cross-linguistic memory effects in parsing.

Efficient representations in WM

For syntactic parsing in NL-Soar, SDM keeps track of partial constituents as the sentence is incrementally structured. Recall from section 2 that SDM consists of a theoretically neutral attribute-value representation. The particular attributes and values used in parsing are incorporated from linguistic theory (in the case of NL-Soar, X-bar phrase structure [Chomsky, 1986], but for expository purposes here we adopt more traditional grammatical relations).

SDM is organized for parsing in the following way. Partial constituents are indexed by the syntactic relations that they may enter into with other constituents (e.g., subject of sentence, object of verb, and so on). SDM is divided into two parts: the *Heads* part indexes constituents by the syntactic relations they may assign. The *Dependents* part indexes constituents by the relations they may receive.

For example, when NL-Soar constructs the prepositional phrase *with the dog*, it must temporarily buffer the NP *the dog* before creating the complete PP. At that moment, the representation in SDM looks like:

(6) HEADS prep-obj: [with]
DEPENDENTS prep-obj: [the dog]
NP-modif: [with]
verb-obj: [the dog]

How should this representation in SDM be limited to ensure efficient processing? The answer comes from theoretical and empirical work on the computational complexity of the recognition match (Tambe, Newell, & Rosenbloom, 1990). This work has identified open, undiscriminated sets in SDM as the most significant source of match expense. An undiscriminated set is simply a set of elements indexed by a single relation or attribute. With such open sets, the recognition match becomes exponentially expensive and therefore psychologically and computationally implausible as the basis for efficient memory retrieval.

Such open sets can occur in any Soar model, and in practice do lead to significant slowdowns in running the simulations. In NL-Soar, undiscriminated sets may be created when multiple constituents are indexed by a single syntactic relation. We can eliminate such open sets by limiting each relation to a small number of nodes. But how many?

It turns out that the minimum number required to parse any natural language is *two* (Lewis, 1996). Any less, and the system would be unable to parse basic structures that relate two propositions, such as sentential complements (*I think that John likes Mary*). We next consider the empirical implications of this severe capacity restriction, then relate it back to general psychological theories of working memory.

Difficult embeddings: The classic memory effect in parsing

The classic short-term memory effect in sentence comprehension is the difficulty of comprehending multiple center embedded relative clauses (Miller & Chomsky, 1963). Single embedded relative clauses such as (7) are quite comprehensible, but double center-embeddings (8) cause severe difficulty:

(7) The cat that the bird chased ran away.

(8) The salmon that the man that the dog chased smoked fell.

Consider now how NL-Soar would parse these structures. To handle a structure such as (7), two noun phrases must be momentarily buffered as potential subjects waiting for their verbs:

(9) potential-subjects: [the cat], [the bird]

However, in (8), by the time *the dog* is read, three potential subjects must be buffered:

(10) potential-subjects: [the salmon], [the man], [the dog]

This exceeds the posited limited of two, and therefore the sentence cannot be parsed.

This simple model, with its uniform limit of two across all syntactic relations, accounts for a wide range of acceptable and unacceptable effects cross-linguistically. The type specificity of the limitation is crucial to the empirical success of the model. To see why, consider the comprehensible Japanese construction in (11) below (Lewis, 1993):

(11) John-wa Bill-ni Mary-ga Sue-ni Bob-o syookai sita to it-ta.
John Bill Mary Sue Bob introduced said

(John said to Bill that Mary introduced Bob to Sue.)

Although such sentences are surely complex, they do not cause the failure associated with (8), despite stacking up five NPs waiting for verbs. The crucial difference, of course, is that (11) requires maintaining no more than two NPs of any particular syntactic function: at most two subjects, two indirect objects, and a direct object.

Lewis (1993) and Lewis (1996) present the complete analysis of this and another four dozen or so constructions from six languages. The data in those papers illustrate the nature of the empirical constraints: a model of syntactic WM must account for the processing difficulty on structures such as (8), while still being powerful enough to handle other complex embeddings , such as (11), which speakers and listeners of a language find acceptable.

The unifying principle: similarity-based interference

How does this model of syntactic working memory relate to existing psychological accounts of working memory or short-term memory? There is an important connection: The model gives rise to similarity-based interference, a key principle underlying other kinds of working memory. The effect is clearly seen from studies demonstrating *retroactive interference*, which occurs when a to-be-remembered stimulus is followed by a set of distracter items that the subject must attend to. The general finding is that distracter items similar in kind to the original stimulus cause forgetting, while dissimilar items are far less disruptive. For example, immediate memory for odor is disrupted by interpolated tasks involving odors, but is unaffected by a distracting verbal task such as counting backwards (Walker & John, 1984). The broad categories for which such interference has been demonstrated include conceptual/semantic (Potter 1976), kinesthetic (Williams et al, 1969), odor (Walker & John, 1984), sign language (Poizner, Bellugi & Tweney, 1981), tone (Deutsch, 1970), verbal (Shiffrin, 1973; Waugh & Norman, 1965), and visual (Logie, Zucco, & Baddeley, 1990). What we are suggesting is that *syntactic* should be added to this list.

Such a suggestion is consistent with the demonstrations of within-category similarity-based interference in several different tasks and modalities, such as the *phonological similarity* effect in standard verbal STM tasks (Baddeley, 1966; Conrad, 1963), and corresponding effects in visual and sign language (Magnussen, Greenlee, Asplund & Dyrnes, 1991; Poizner et al, 1981). We are claiming that there is also a “syntactic similarity effect”, which shows up as difficulty with center-embeddings. A consistent, emerging picture is one that characterizes working memory in terms of specific coding schemes, with severe limitations on the ability to represent items that are coded similarly.

Similarity-based interference, in one form or another, is part of a number of current theoretical frameworks, including Baddeley & Logie (this volume), Cowan (this volume), and O’Reilly, Braver & Cohen (this volume). Connectionist approaches such as the latter are particularly interesting in this respect, since network models exhibit similarity-based interference as an intrinsic property of their representation mechanisms.

7. Unitary vs Non-Unitary WM (Q 3)

As we have seen, Soar postulates a dynamic memory, SDM, which holds all the information in declarative form pertaining to the task in hand. As discussed in the previous section, however, although SDM is a single memory in regard to its function within the Soar architecture, the existence

within it of different encodings and representations (arising from functional considerations) can give rise to what are effectively multiple memories. The processing of language, for example, involves information in specialised representations encoding syntactic and semantic relations between the various words, while the processing of, say, spatial information involves different, specialised representations for the spatial relationships between objects. To the extent that linguistic information is processed by production rules which are sensitive to linguistic data encoded in the linguistic representation and which produce further information in that representation, while spatial information is processed by rules sensitive to the spatial representation and which produce further information in that representation, so the linguistic and the spatial data can be regarded as residing in different memories. These multiple memories may be active for the same task. For example, a person processing sentences asserting *A is to the left of B* and *C is to the right of B* may well have both a representation of the information reflecting its original linguistic structure as well as a more directly spatial encoding of the configuration being described.

Soar is neutral as to whether such functionally separable memories are or are not localised, i.e. stored at different physical locations, in the brain. Soar does, however, offer a couple of constraints on the extent to which the multiple memories can conform to an extreme version of modularity (Fodor, 1983). First is the observation that many tasks require transfer of information between and integration across the different representations. For example, to construct a spatial-like representation from sentences such as *A is to the left of B* requires rules that can translate information from one representation to the other, and therefore "know about" both. The second constraint comes from the integrative nature of Soar's learning mechanism. Just as a Soar model of natural language processing acquires rules whose conditions combine lexical, syntactic, semantic, and pragmatic information (Lewis, 1993), so other tasks which involve deliberate processing (i.e. processing in a substate) of information held in different representations lead to the learning of rules that integrate over the different representations. There are therefore functional limits on just how independent the multiple memories can be.

8. Biological Considerations (Q 8)

The main way in which Soar contacts its implementational roots concerns timing. Newell (1990, 1992) employs a biologically-based argument to identify the different timescales at which Soar operates. Newell uses the argument to squeeze the durations of Soar's processes between lower and upper bounds, in order to derive estimates such as the time taken to execute a cognitive operator uninterrupted by an impasse. For example, an increasing number of studies show that a figure of around 50 msec provides a consistent, good approximate fit to human performance across a variety of domains (e.g. Wiesmeyer & Laird, 1993; Lewis, 1993). This argument is of great theoretical importance, since it makes Soar the only major psychological theory to yield *absolute* (even though approximate) temporal predictions.

Beyond those temporal issues, Soar is largely neutral as to its biological implementation. Even the apparent commitment to using a production system is not an essential feature of Soar. The role of the existing production system is to provide Soar with a long-term, content-addressed associative memory. In principle, one could replace the production system by any other system for a content-addressed memory, such as one based on connectionist principles (Cho, Rosenbloom & Dolan, 1991), and leave the architectural aspects of Soar intact. In practice, such a substitution is likely to lead to modifying the architecture in interesting ways.

9. The Relationship of WM to Attention and Consciousness (Q 7)

From our perspective, *attention* and *consciousness* are very different issues, and we treat them separately.

Attention

According to Soar's viewpoint, *attention* is not a process separate from the rest of cognition, but rather a label for a collection of phenomena that emerge when a system with potential for a high degree of parallelism has to generate serial behaviour. Whenever Soar has to deal with one object, or a small set of objects, out of the many that could be dealt with, then Soar is said to be *attending* to that object or set. As with so much else in Soar, attention is closely tied to the serial bottleneck imposed by the selection of one operator at a time. Suppose that Soar is holding three words in SDM, and the task is to check that each of them is the name of an animal. If we assume, as seems reasonable, that the relevant operators deal with only one word at a time, then Soar can propose in parallel three operators of type *check-for-animal-name*, one for each of the words. But the operators have to be selected and implemented serially, with the result that Soar "pays attention to" each word in turn. The Soar model most explicitly dealing with attentional phenomena is that of Wiesmeyer (1992; Wiesmeyer & Laird, 1993), in which a single model and set of assumptions provides a zero-parameter quantitative and qualitative fit to a range of phenomena in the psychology of covert visual attention.

Consciousness

The term *consciousness* has almost as many different meanings as there are commentators. Soar takes a stand mainly on one particular sense, that of conscious *awareness*. The natural way to read off a micro-theory of awareness from the Soar architecture is functionally: Soar is aware of some item *X* if its behavior can be made to depend on *X*. Each item in SDM can potentially — given appropriate knowledge (i.e. production rules) — come to determine Soar's course of processing, and hence "enter awareness". In short, Soar is aware of the contents of SDM, which as we have seen, is a subset of the contents of working memory. Whether or not it can produce a verbal report of those contents is another question, which depends upon the representations involved, the cognitive demands of concurrent activity, and so on. Although one can imagine refinements to this position — for example, proposed but unselected operators probably remain outside of awareness, as perhaps do even selected operators —the approach seems viable, and indeed is very close to the position adopted by Ericsson & Simon (1980, 1984) in their treatment of the information available to someone in the context of a psychological experiment.

10. Conclusion

In this chapter we have considered working memory from the point of view of Soar, a broad cognitive architecture primarily motivated by functional concerns. At first blush, it was not clear that such an architecture would offer much to the theoretical study of WM, since, in its original formulation, it did not incorporate any constraints on WM capacity. However, the study of WM from this functional point of view has provided answers to many of the designated questions, as summarised in Table 1.

Table 1. Brief summary of answers to the eight designated questions.

---

(1) Basic mechanisms and representations in working memory

& (2) The control and regulation of working memory

(These first two questions are answered together.) Soar is a cognitive architecture of broad scope, which focuses on the functional capabilities needed to support performance in a range of cognitive tasks. The Soar architecture has no separate "WM mechanism", in the sense of components aimed specifically at accounting for memory phenomena. Soar is a production system architecture, with two main memories: a long-term production memory, storing permanent knowledge, and a dynamic memory which holds state information pertaining to the current task. The functions of working memory are distributed across both these memories. Processing occurs by cyclically choosing an operator to apply to the current state, thereby transforming it to a new state. Whenever that process is blocked, Soar sees an "impasse" and sets up a substate, processing of which is intended to produce information to allow processing of the original state to resume. New production rules are acquired whenever processing of a substate generates information for the original state.

(3) The unitary vs non-unitary nature of working memory

Although Soar postulates a single dynamic memory, the existence of different encodings (arising from functional considerations) can give rise to what are effectively "multiple memories". Soar is neutral as to whether such functionally separable memories are localised in the brain. There is growing evidence that to extend Soar's correspondence to human memory, it may be necessary to supplement Soar's main dynamic memory with a number of short-term modality-specific stores, of which a phonological store is one.

(4) The nature of working memory limitations

We make two points. First, even in a cognitive architecture with an unbounded dynamic memory, WM limitations can arise on functional grounds. Where such functional accounts exist, they render the usual kinds of resource-based explanations unnecessary. Second, Soar is compatible with certain kinds of capacity limitations. In particular, a constraint that dynamic memory can hold at most two items of the same "type" yields a coherent explanation for many psycholinguistic phenomena in sentence comprehension, and embodies a general cognitive principle—similarity-based interference—operative across multiple kinds of working memory contents.

(5) The role of working memory in complex cognitive activities

Owing to its production system architecture, Soar is strong at modelling the performance of complex cognitive tasks. By making recognitional use of information in long-term memory acquired as a by-product of earlier task performance, Soar shows how tasks requiring search of a large problem space or the accumulation of large amounts of information can be performed within the constraints of a dynamic memory of bounded size.

(6) The relationship of working memory to long-term memory and knowledge

Soar's working memory system *includes* long-term memory as one of its component mechanisms. Long-term memory serves the function of a working memory whenever it encodes, via learning, partial or intermediate products of computation in service of some task. Soar therefore has close links to other approaches which emphasise the involvement of long-term memory in WM, and may be able to offer a computational process model for Ericsson & Kintsch's "long-term WM". The analysis shows that WM cannot coherently be studied independently of long-term memory and while ignoring learning.

(7) The relationship of working memory to attention and consciousness

Attentional phenomena are "real", but attention is not a separate "thing". Rather, it labels a collection of effects that emerge when a system with potential for a high degree of parallelism has to generate serial behaviour. With regard to consciousness, Soar is "aware" of what it can respond to, i.e. of the contents of its dynamic memory. Whether or not it can produce a verbal report of those contents depends upon a variety of other factors.

(8) The biological implementation of working memory

Newell (1990) employs a biologically-based argument to identify the different timescales at which Soar operates. This argument is of great theoretical importance, since it makes Soar the only major psychological theory to yield absolute (even though approximate) temporal predictions. Beyond that, Soar is neutral as to its biological implementation. In principle, one could replace the production system by any other implementation of a content-addressed associative memory, such as a connectionist network.

Soar's approach to issues of WM has yielded three novel contributions:

1. One of the distinctive contributions of Soar is to offer a detailed account of how complex human cognition can work even within a small working memory. There are now a number of Soar models that illustrate this point, and all rely on the same principle: memory-demanding tasks are supported by relying on Soar's learning mechanism to produce a recognition memory during task performance. In most of the tasks studied so far, this recognition memory emerges as a by-product of normal task performance. The recognition memory is used reconstructively by relying on internally generated cues, or externally supplied (i.e., "situated") cues. These models go a long way toward helping resolve one of the troubling paradoxes about cognitive models of complex tasks (particularly production system models): most such models rely on implausibly large dynamic memories to support performance. The Soar models discussed in this chapter provide a concrete set of mechanisms that clearly demonstrate that such oversize memories are not needed. The resulting theory of performance has close ties to the ideas surrounding long-term working memory (Ericsson & Kintsch, 1995), provides a candidate set of mechanisms for realizing those ideas, and extends the relevant domains to any task requiring significant temporary memory storage—not just overlearned expert performance.

   This is good news for both traditional working memory theorists concerned with capacity limitations, and theorists concerned with complex task performance. For the former, it means that it may be possible to incorporate even severely limited models of working memory within a functional architecture; for the latter, it means that it should be possible to build new models (as

well as reconstruct existing models) of complex cognition in such a way that the demands on working memory are minimal.

2. The Soar models of reasoning and language demonstrate that some phenomena which are attributed to capacity limitations can arise from purely functional mechanisms or differences in knowledge or skill. In particular, we saw how Soar's control structure can yield performance limitations when it leads to the application of automatized skill in inappropriate situations. The Soar theory also includes an account of how certain skill differences may actually arise as a function of experience, providing an explanation of both the locus and genesis of individual differences that might otherwise be assumed to be capacity-based.

3. Although Soar as originally formulated had no WM capacity limits, concern for efficient processing (in particular, retrieval from the recognition memory) led to a constraint on SDM that yields similarity-based interference. In short, similarly-coded items interfere more than dissimilar items. The application of this restriction to the domain of syntactic processing (along with an assumption about the severe limit on the number of similar linguistic items that can be maintained—the "Magical Number Two"), led to a simple model of difficult and acceptable embeddings that account for a wide range of processing effects across multiple languages. Similarity-based interference has been demonstrated in many different tasks and modalities. The various kinds of codings (phonological, visual, etc.) give rise to specialized memories, but these specializations emerge from, and are governed by, the same general cognitive principle.

By focusing on learning, long-term memory, and functionality, this story has strayed outside the traditional concerns of working memory research—but for good reasons. Though we are obviously in the early stages of this research, a couple of methodological lessons should be clear, even if particular theoretical aspects of Soar remain contentious. The first lesson is that we must be careful not to consider working memory in isolation from long-term memory: they are designed to work together (and with the external environment) in the performance of complex tasks. The second lesson is that we cannot separate learning from performance— even in what is conventionally regarded as pure "performance" tasks, learning can play an important role. Such lessons should not be cause for scientific pessimism because they defeat certain divide-and-conquer approaches to psychological investigation. Rather, they should encourage us to work harder on developing and empirically distinguishing precise unified architectures that explain how the different aspects of cognition work together.

References

Altmann, E. M. (1996) Episodic memory for external information. PhD thesis, School of Computer Science, Carnegie Mellon University. CMU-CS-96-167.

Altmann, E. M. & John, B. E. (in press) Episodic long-term working memory. *Cognitive Science*, in press.

Altmann, E. M., Larkin, J. H. & John, B. E. (1995) Display navigation by an expert programmer: A preliminary model of memory. In I. R. Katz, R. Mack & L. Marks (Eds), *Proceedings of CHI'95: Human Factors in Computing Systems*, 3-10. ACM Press.

Anderson, J. R. (1983) *The Architecture of Cognition.* Cambridge, MA: Harvard University Press.

Anderson, J. R. (1993) *Rules of the Mind*. Hillsdale, NJ: Lawrence Erlbaum.

Anderson, J. R. & Bower, G. H. (1972) Recognition and retrieval processes in free recall. *Psychological Review, 79*, 97-123.

Atkinson, R. C. & Shiffrin, R. M. (1968) Human memory: A proposed system and its control processes. In K. W. Spence (Ed), *The Psychology of Learning and Motivation: Advances in Research and Theory*, Volume 2, 89-195. New York: Academic Press.

Baddeley, A. D. (1966) Short-term memory for word sequences as a function of acoustic, semantic, and formal similarity. *Quarterly Journal of Experimental Psychology, 18*, 362-365.

Baddeley, A. D. (1986) *Working Memory.* Oxford University Press.

Baddeley, A. D. (1996) Exploring the central executive. *Quarterly Journal of Experimental Psychology, 49A*, 5-28.

Baddeley, A. D. & Hitch, G. J. (1974) Working memory. In G. A. Bower (Ed), *Recent Advances in Learning and Motivation*, Volume 8. New York: Academic Press.

Bahrick, H. P. (1970) Two-phase model for prompted recall. *Psychological Review, 77*, 215-222.

Broadbent, D. E. (1993) Comparison with human experiments. In Broadbent, D. E. (Ed), *The Simulation of Human Intelligence.* Oxford, UK: Blackwell.

Cho, B., Rosenbloom, P. S. & Dolan, C. P. (1991) Neuro-Soar: A neural network architecture for goal-oriented behavior. In *Proceedings of the 13th Annual Conference of the Cognitive Science Society*. Chicago.

Chomsky, N. (1986) *Barriers.* Cambridge, MA: MIT Press.

Congdon, C. B. & Laird, J. E. (1995) The Soar user's manual: Version 7. Department of Electrical Engineering and Computer Science, University of Michigan. Ann Arbor, MI.

Conrad, R. (1963) Acoustic confusions and memory span for words. *Nature, 197*, 1029-1030.

Cowper, E. A. (1976) Constraints on sentence complexity: A model for syntactic processing. PhD thesis, Brown University.

Deutsch, D. (1970) Tones and numbers. *Science, 168*, 1604-1605.

Ericsson, K. A. & Simon, H. A. (1980) Verbal reports as data. *Psychological Review, 87*, 215-251.

Ericsson, K. A. & Simon, H. A. (1984) *Protocol Analysis: Verbal Reports as Data*. Cambnridge, MA: MIT Press.

Ericsson, K. A. & Kintsch, W. (1995) Long-term working memory. *Psychological Review, 102*, 211-245.

Ferriera, F. & Clifton, C. (1986) The independence of syntactic processing. *Journal of Memory and Language, 25*, 348-368.

Franzke, M. (1995) Turning research into practice: Characteristics of display-based interaction. In I. R. Katz, R. Mack & L. Marks (Eds), *Proceedings of CHI'95: Human Factors in Computing Systems*, 421-428. ACM Press.

Gruneberg, M. M., Morris, P. E. & Sykes, R. N. (Eds) (1988) *Practical Aspects of Memory: Current Research and Issues. Volume 1: Memory in everyday life.* Wiley.

Howes, A. (1993) Recognition-based problem solving. *Proceedings of the Fifteenth Annual Meeting of the Cognitive Science Society.* Boulder, Colorado, 551-556.

Howes, A. (1994) A model of the acquisition of menu knowledge by exploration. In B. Adelson, S. Dumais & J. R. Olson (Eds), *Proceedings of CHI'94: Human Factors in Computing Systems*, 445-451. New York: ACM Press.

Howes, A. & Young, R. M. (1996) Learning consistent, interactive and meaningful device methods: A computational model. *Cognitive Science, 20*, 301-356.

Howes, A. & Young, R. M. (in press) The role of cognitive architecture in modelling the user: Soar's learning mechanism. *Human-Computer Interaction*, in press.

Huffman, S. B. (1994) Instructable autonomous agents. PhD dissertation, Department of Electrical Engineering and Computer Science, University of Michigan. Ann Arbor, MI. Report CSE-TR-193-94.

Huffman, S. B. & Laird, J. E. (1995) Flexibly instructable agents. *Journal of Artificial Intelligence Research, 3*, 271-324.

Johnson-Laird, P. N. (1983) *Mental Models.* Cambridge University Press.

Johnson-Laird, P. N. (1988) Reasoning by rule or model? *Proceedings of the Tenth Annual Conference of the Cognitive Science Society*, 765-771. Montreal.

Just, M. A. & Carpenter, P. A. (1992) A capacity theory of comprehension: Individual differences in working memory. *Psychological Review, 99*, 122-149.

Kintsch, W. (1970) Models for free recall and recognition. In D. A. Norman (Ed.), *Models of Human Memory*. New York: Academic Press.

Laird, J. E., Rosenbloom, P. S. & Newell, A. (1986) *Universal Subgoaling and Chunking: The Automatic Generation and Learning of Goal Hierarchies.* Kluwer.

Laird, J. E., Newell, A. & Rosenbloom, P. S. (1987) Soar: An architecture for general intelligence. *Artificial Intelligence, 33*, 1-64.

Larkin, J. H. (1989) Display-based problem solving. In D. Klahr, & K. Kotovsky (Eds.), *Complex information processing: The impact of Herbert A. Simon*. Hillsdale, NJ: Erlbaum.

Lehman, J. F., Laird, J. E. & Rosenbloom, P. S. (1996) A gentle introduction to Soar, an architecture for human cognition. In S. Sternberg & D. Scarborough (Eds), *Invitation to Cognitive Science, Volume 4*. Cambridge, MA: MIT Press.

Lewis, R. L. (1993) An architecturally-based theory of human sentence comprehension. PhD thesis, School of Computer Science, Carnegie Mellon University. CMU-CS-93-226.

Lewis, R. L. (1996) Interference in short-term memory: The magical number two (or three) in sentence processing. *Journal of Psycholinguistic Research, 25,* 93-115.

Lewis, R. L. (1997a) Leaping off the garden path: Reanalysis and limited repair parsing. In J. Fodor & F. Ferreira (Eds), *Reanalysis in Sentence Processing*. Boston: Kluwer.

Lewis, R. L. (1997b) Specifying architectures for language processing: Process, control, and memory in parsing and interpretation. In M. Crocker, M. Pickering & C. Clifton (Eds), *Architectures and Mechanisms for Language Processing*. Cambridge University Press.

Lewis, R. L., Newell, A. & Polk, T. (1989) Toward a Soar theory of taking instructions for immediate reasoning tasks. *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society,* 514-521. Ann Arbor, Michigan.

Logie, R., Zucco, G. M. & Baddeley, A. D. (1990) Interference with visual short-term memory. *Acta Psychologica, 75*, 55-74.

MacDonald, M., Pearlmutter, N. & Seidenberg, M. (1994) The lexical nature of syntactic ambiguity resolution. *Psychological Review, 101,* 676-703.

Magnussen, S., Greenlee, M. W., Asplund, R. & Dyrnes, S. (1991) Stimulus-specific mechanisms of visual short-term memory. *Vision Research, 31*, 1213-1219.

Mayes, J. T., Draper, S. W., McGregor, A. M. & Oatley, K. (1988) Information flow in a user interface: The effect of experience and context on the recall of MacWrite screens. In D. M. Jones & R. Winder (Eds), *People and Computers IV*, 275-289. Cambridge University Press.

Miller, C. S. & Laird, J. E. (1996) Accounting for graded performance within a discrete search framework. *Cognitive Science, 20*, 499-537.

Miller, G. A. (1956) The magic number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review, 63*, 81-97.

Miller, G. A. & Chomsky, N. (1963) Finitary models of language users. In Luce, D. R., Bush, R. R. & Galanter, E. (Eds.), *Handbook of Mathematical Psychology, Vol. II.* New York: Wiley.

Miller, G. A., Galanter, E. & Pribram, K. (1960) *Plans and the Structure of Behavior.* New York: Holt, Rinehart & Winston.

Murdock, B. B. (1963) Short-term memory and paired-associate learning. *Journal of Verbal Learning and Verbal Behavior, 2*, 320-328.

Neisser, U. (1982) *Memory Observed: Remembering in Natural Contexts*. San Francisco: W. H. Freeman.

Neisser, U. (1985) The role of theory in the ecological study of memory: comment on Bruce. *Journal of Experimental Psychology: General*, 114, 272-276.

Newell, A. (1990) *Unified Theories of Cognition.* Harvard University Press.

Newell, A. (1992) Précis of Unified Theories of Cognition. *Behavioral and Brain Sciences, 15*, 425-492.

Newell, A. & Simon, H. A. (1972) *Human Problem Solving.* Englewood Cliffs, NJ: Prentice Hall.

Payne, S. J. (1991) Display-based action at the user interface. *International Journal of Man-Machine Studies, 35*, 275-289.

Polk, T. A. & Newell, A. (1995) Deduction as verbal reasoning. *Psychological Review, 102*, 533-566.

Poizner, H. , Bellugi, U. & Tweney, R. D. (1981) Processing of formational, semantic, and iconic information in American Sign Language. *Journal of Experimental Psychology: Human Perception and Performance, 7*, 1146-1159.

Potter, M. (1976) Short-term conceptual memory for pictures. *Journal of Experimental Psychology: Human Learning and Memory, 2*, 509-522.

Rieman, J. (1994) Learning strategies and exploratory behavior of interactive computer users. Ph.D. thesis, University of Colorado at Boulder, Department of Computer Science Technical Report: CU-CS-723-94.

Rieman, J. (1996) A field study of exploratory learning strategies. *ACM Transactions on Computer-Human Interaction, 3*, 189-218.

Rieman, J., Young, R. M. & Howes, A. (1996) A dual-space model of iteratively deepening exploratory learning. *International Journal of Human-Computer Studies, 44*, 743-775.

Ritter, F. E. & Young, R. M. (on line) Psychological Soar Tutorial. Available on the World Wide Web at http://www.psychology.nottingham.ac.uk/staff/ritter/pst.html .

Shiffrin, R. M. (1973) Information persistence in short-term memory. *Journal of Experimental Psychology, 100*, 39-49.

Tambe, M., Johnson, W. L., Jones, R. M., Koss, F., Laird, J. E., Rosenbloom, P. S. & Schwamb, K. (1995) Intelligent agents for interactive simulation environments. *AI Magazine, 16*, 15-40.

Tambe, M., Newell, A. & Rosenbloom, P. S. (1990) The problem of expensive chunks and its solution by restricting expressiveness. *Machine Learning, 5*, 299-348.

Trueswell, J. C., Tanenhaus, M. K., & Garnsey, S. M. (1994) Semantic influences in parsing: Use of thematic role information in syntactic ambiguity resolution. *Journal of Memory and Language, 33*, 285-318.

Vera, A. H., Lewis, R. L., Lerch, F. J. (1993) Situated decision-making and recognition-based learning: Applying symbolic theories to interactive tasks. *Proceedings of the 15th Annual Conference of the Cognitive Science Society*, 84-95. Boulder, Colorado.

Walker, H. A. & John, E. E. (1984) Interference and facilitation in short-term memory for odors. *Perception & Psychophysics, 36*, 508-514.

Waugh, N. C. & Norman, D. A. (1965) Primary memory. *Psychological Review, 72*, 89-104.

Wiesmeyer, M. D. (1992) An operator-based model of human covert visual attention. PhD thesis, University of Michigan.

Wiesmeyer, M. & Laird, J. E. (1993) NOVA, covert attention explored through unified theories of cognition. In *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society*, 102-107. Boulder, Colorado.

Williams, H. L., Beaver, W. S., Spence, M. T., and Rundell, O. H. (1969) Digital and kinesthetic memory with interpolated information processing. *Journal of Experimental Psychology, 80*, 530-536.