Cognitive Science 26 (2002) 39–83

COGNITIVE SCIENCE

http://www.elsevier.com/locate/cogsci

Memory for goals: an activation-based model

Erik M. Altmann[a,*], J. Gregory Trafton[b]

[a] *Michigan State University, East Lansing, MI 48824, USA*
[b] *Naval Research Laboratory, 4555 Overlook Avenue SW, Washington DC 20375, USA*

Abstract

Goal-directed cognition is often discussed in terms of specialized memory structures like the "goal stack." The *goal-activation* model presented here analyzes goal-directed cognition in terms of the general memory constructs of activation and associative priming. The model embodies three predictive constraints: (1) the interference level, which arises from residual memory for old goals; (1) the strengthening constraint, which makes predictions about time to encode a new goal; and (3) the priming constraint, which makes predictions about the role of cues in retrieving pending goals. These constraints are formulated algebraically and tested through simulation of latency and error data from the Tower of Hanoi, a means-ends puzzle that depends heavily on suspension and resumption of goals. Implications of the model for understanding intention superiority, postcompletion error, and effects of task interruption are discussed. © 2002 Cognitive Science Society, Inc. All rights reserved.

Keywords: Memory; Problem-solving; Cognitive simulation; Goals; Towers of Hanoi; Prospective memory

1. Introduction

When people plan, react to emergencies or other interruptions, or make mental notes to do things in the future, an important process involved is the cognitive management of goals. We use the term "goal" to refer to a mental representation of an intention to accomplish a task, achieve some specific state of the world, or take some mental or physical action (contrasting with use of "goal" to mean a performance criterion; e.g., Carver & Scheier, 1998; Heath, Larrick, & Wu, 1999). Very often, a goal must be suspended or set aside temporarily and then resumed later. Suspending a goal might be necessary, for example, if it requires subgoals to be achieved first, as in hierarchical problem solving (Miller, Galanter & Pribram,

* Corresponding author. Tel.: +1-517-353-4406; fax: +1-517-353-1652.
E-mail addresses: ema@msu.edu (E.M. Altmann), trafton@itd.nrl.navy.mil (J.G. Trafton).

PII: S0364-0213(01)00058-1

1960) and means-ends analysis (Ernst & Newell, 1969; Newell & Simon, 1972). Alternatively, progress on a goal might be blocked by the state of the world, so one may benefit from turning attention to some other task until the environment changes (e.g., Patalano & Seifert, 1997). Or, one might be interrupted in the middle of a task, prompting the formulation of an intention to resume the task later (e.g., Marsh, Hicks & Bink, 1998; Patalano & Seifert, 1994). Each of these scenarios requires the cognitive system at some point to resume a goal that had to be suspended, which in turn involves thinking back to a previous state of the world. The better the system can remember how far it had progressed toward achieving the pending goal, the more accurately and efficiently it can resume the goal. Indeed, properly resuming a task at the right step, without taking redundant actions but also without skipping steps, can be a matter of life and death in domains like aviation (Latorella, 1996).

The question addressed in this article is whether an analysis in terms of memory theory—particularly the constructs of activation and associative priming—can help us understand how the cognitive system remembers its goals. Our hope is to identify cognitive and environmental constraints on how we store goals in memory and retrieve them later. We begin with a brief history of theorizing about cognitive goal structures, to show how and why current theory retains the assumption that goal memory is structurally different from and superior to memory for other facts and events. The review prepares us to argue that this assumption is misleading and needs to be updated to make accurate predictions about goal-directed behavior.

We then develop the goal-activation model, which identifies three constraints on goal-directed behavior: the interference level, the strengthening constraint, and the priming constraint. The interference level is a mental "clutter" of residual memory for old goals, which is simply an instance of interference generally (e.g., Keppel, Postman & Zavortink, 1968; Waugh & Norman, 1965). The strengthening constraint says that the activation of a new goal must be increased to overcome proactive interference. This strengthening process is not instantaneous and therefore predicts a behavioral time lag to encode a new goal. Importantly, this lag is functionally bounded from above, because excessive strengthening raises the interference level by making that goal more likely to capture behavior in the future, when it is no longer the target. The priming constraint says that a suspended (pending) goal can only be retrieved with help from priming from some associated cue, to overcome retroactive interference from intervening goals. The priming constraint has implications for the structure of the task environment, in that the cue to which a goal is associatively linked must be available both when the goal is suspended (in order to create the link) and when the goal is resumed (in order to prime the target).

After an introduction to the goal-activation model, we describe a computational simulation that fits and predicts behavioral data from the Tower of Hanoi puzzle. This puzzle is a means-ends task that exercises the cognitive system's ability to remember old goals, and that affords a chance to test our predictions concerning time lag to encode a goal and the presence of appropriate cues in the task environment. In the General Discussion, we re-examine empirical findings—the intention-superiority effect, in particular—previously taken to support the notion that goals are stored in memory in a special state. We also use the goal-activation model to ask how the cognitive system responds to commonplace interruptions like the telephone ringing, and to suggest directions for further research in this important applied domain. Appendix A develops an algebraic version of the model and works through an example of how to use it to predict the amount of time needed to encode a goal given a particular tempo of goals arising over time. Appendix B elaborates on the computational simulation, describing processes that may be relevant to memory simulations more generally.

2. A special goal memory: from simplifying assumption to theoretical construct

Goal-directed cognition is often discussed in terms of cognitive structures that happen to mirror the goal-related requirements of the task environment. For example, when a task is hierarchical and requires goal decomposition, the system's goals are often assumed to reside in a stack, a data structure that provides an appropriate kind of last-in, first-out access. The argument is typically a functional one—because the system can perform hierarchical tasks, it must contain data structures that support the necessary access patterns for goals. However, empirical evidence for such goal structures is lacking, and our view is that better predictions and more parsimonious models of memory for goals will follow if we look to basic cognitive constraints, like those that shape memory generally. Before we examine these cognitive constraints, we briefly review the history behind the notion that goals are stored in a different, special memory than are other kinds of facts and events. We will argue, first, that this notion is an artifact of tactical decisions in early cognitive research, and second, that it is high time to update our assumptions, because the construct of a special goal memory has taken on a life of its own in the psychological literature.

The functional argument that special goal structures exist because the system needs them follows from the blurring of two distinctions in early cognitive research. The first distinction is that between control flow—the sequencing of physical or mental steps—and the mental storage structures that support it. For example, if a task is too complex to perform directly, means-ends analysis (Ernst & Newell, 1969; Newell & Simon, 1972) is a control structure that can be used to decompose it into subgoals. If these subgoals are also too complex to achieve directly, they must be decomposed in turn. Eventually the system reaches a subgoal tractable enough to achieve directly, then pops back to see what progress can be made on the superordinate goal. This "popping back" operation depends critically on the supergoal being readily available in memory. If control flow is the target of analysis, rather than memory, it makes sense methodologically to simplify the problem and assume that supergoals are indeed reliably available when the executive system needs to retrieve them.

The second distinction is that between task constraints and cognitive constraints. Both kinds of constraint must be met by a system performing a task, but the meeting of task constraints is more easily observed—the analyst or observer need only monitor whether the system's performance is successful. In contrast, the influence of cognitive constraints can only be observed indirectly, for example with additive-factors logic (Sternberg, 1998). In particular, successful task performance by itself proves little about such internal constraints, beyond demonstrating the sufficiency of whatever underlying processes are at work (Simon, 1975). Thus, for example, the working memory capacity of the system may be less than ideal for solving complex problems, and yet the system can manage if it can bring processes to bear that compensate for inadequate (internal) memory. Again, if the target of analysis is how the structure of a complex task influences behavior, it may make sense simply to ignore such internal constraints as memory and defer their analysis to a later time.

Early studies of higher-level cognition and intelligence, where goals first appeared on the analytical horizon, were indeed focused more on the flow of behavior in response to task structure than on human memory and its constraints. For example, hierarchical control was the focus one of the first studies of the cognitive revolution, but the approach was purely analytical and addressed no human data that might have indicated the role of memory limitations (Miller et al., 1960). Similarly, when the power and generality of means-ends analysis were initially established, this was done through computational simulation of performance, intentionally without close attention to cognitive constraints (Ernst & Newell, 1969). When human behavior was examined for evidence of means-ends analysis, the protocol data used were too coarse-grained to provide much insight into low-level memory constraints (Newell & Simon, 1972). Thus, the door was open to a default view of goal memory as structurally adapted to the complex control flow required by problem solving and other kinds of higher-level cognition.

As we've suggested, one structure well suited to hierarchical control in general and means-ends analysis in particular is the stack. In a stack, elements are ordered by age, with the newest element on top (the most accessible position) and the oldest on the bottom (the least accessible position). Such an organization is widely used in computer science, and often around the home and office. In cognitive architectures that incorporate a goal stack, the newest goal is the one that directs behavior (Anderson & Lebiere, 1998; Newell, 1990). If this goal must be decomposed because it is too large to achieve directly, its subgoals are "pushed" on the stack, effectively suspending the goal and shifting control to one of its subgoals. When a subgoal is achieved, it is "popped" and the system returns control to the supergoal. Thus, the goal stack delivers the right goal at the right time during means-ends analysis. Of course, just because the stack is sufficient to meet this need does not make it necessary. A less suitable structure for storing goals will do, as long as the system can draw on processes that make up the functional difference. The stack allows the analyst to postpone questions about what those compensatory processes might be and how they work—questions that we address directly, here.

A stack-like goal memory was tacitly assumed in a number of influential studies of higher-level control. The original analysis of hierarchical plans (Miller et al., 1960) assumed that all goals relevant to the current activity were also available to direct behavior. For example, abstract goals (e.g., being a good citizen) were taken to direct actions (e.g., giving alms to the poor) concurrently with specific goals (e.g., helping a particular person). If the value of a candidate action must be measured against all relevant goals before the action is selected, all such goals must be available to the evaluation process. At least one computational implementation of the goal stack reflects precisely this kind of parallelism (Newell, 1990), which also lives on in verbal theories of goals as regulatory constructs (Carver & Scheier, 1998; Powers, 1973). Similarly, the most ambitious of the early studies of means-ends analysis simply assumed that people could backtrack to old mental states (Newell & Simon, 1972), without focusing on the details of how such states were retrieved or reconstructed. A reader might be forgiven for coming away with the view that backtracking was supported by some kind of special cognitive structure.

Special goal structures became a more explicit assumption with the development of the ACT* (Anderson, 1983) and Soar (Newell, 1990) cognitive architectures. In ACT*, for example, goals were sources of activation without requiring active maintenance, and were linked associatively to other goals to which they were related by task constraints. This meant that returning control to a supergoal, for example, was a reliable operation, because the subgoal simply pointed to it. Soar contains no notion of activation—items are either in working memory (WM) or they are not—but the goal stack is part of WM and hence its contents are always immediately available (Young & Lewis, 1999). The real contribution of such goal structures, especially as implemented in cognitive architectures, was methodological—they allowed the analyst to divide and conquer. Complex problem solving behaviors could be simulated without having to implement a full-blown memory theory.

Like any psychological theory, the goal stack has also had evidence adduced against it. The stack predicts a pervasive tendency to select goals in a last-in, first-out (LIFO) order, but goal-selection order is highly idiosyncratic in non-LIFO tasks (VanLehn, Ball & Kowalski, 1989), and the stack is difficult to "patch" in a way that explains systematic forgetting of certain kinds of goals (Byrne & Bovair, 1997). Contrary to the functional argument for a stack, behavior that seems to be governed by serial goal selection can sometimes be attributed to perceptual-motor constraints (Meyer & Kieras, 1997), and where a LIFO goal ordering is necessary it can often be reconstructed from the environment (VanLehn & Ball, 1991). Finally, Anderson and colleagues have raised their own doubts about the goal stack[1] and have begun to tease apart goal management processes experimentally (Anderson & Douglass, 2001).

Despite its problems, the notion of a specialized memory for goals has acquired the status of a theoretical construct in the psychological literature. Here we touch on two high-profile examples, to which we return at the end of the article with our own model in hand. The first involves a recent lead article in the *Journal of Experimental Psychology: General* that cites special goal memory in ACT theory as support for the existence of active inhibitory processes. The question raised by Mayr and Keele (2000) is whether an intention (or goal) to perform a task is inhibited once an intention to perform a different task supersedes it. The pro-inhibition argument (see Conway, 1999, for an introduction to the debate) is that because intentions are stored in a special state, active inhibition is required to remove them from working memory. Mayr and Keele (2000) suggest that "high-level control settings have a tendency for prolonged or even self-sustained activation," and that because of this hyperactive state, "ACT (Anderson, 1983, 1993) contains a mechanism for eliminating goals from working memory ('goal popping')" (p. 6). That is, goals are so active that they need to be actively inhibited with a "popping" mechanism.

The second example involves the *intention superiority* effect. The finding is that memory for an action script is better than memory for a "neutral" script with no associated action, and also better than an "observe" script, where the action is to be observed rather than carried out. Goschke and Kuhl (1993) set the context for this finding again with reference to goals in ACT*:

> According to Lewin (1926), intentions produce a persisting “task tension” leading to superior recall of incomplete activities. More recently, a related assumption has been formulated by Anderson (1983) within a cognitive framework. In Anderson’s adaptive control of thought (ACT*) model, so-called “goal nodes” are conceived of as sources “of high and constant activation” (p. 156) and form the only elements in working memory “that sustain activation without rehearsal.” (Goschke & Kuhl, 1993, p. 118.)

In follow-up studies Marsh, Hicks and Bink (1998) note that “goal nodes are the only elements of working memory in ACT* that do not need rehearsal to sustain activation.” Partly on these grounds, Marsh et al. (1998) rule out the possibility that intention superiority is caused by rehearsal or recoding, and their subsequent experiments are meant to provide “convergent evidence that intended activities reside in memory with some special status.” Here, then, the assumption that goal memory is special not only shaped the interpretation of empirical results, but also decisions about what questions to ask in the first place. This special-status account of intention superiority has been consummated in a computational model in which the appropriate intention persists, immune from forgetting, until the associated actions are carried out (Lebiere & Lee, 2001).

In summary, the goal stack was a response to the original challenges of cognitivism—it was necessary first to trace complex mental behavior, for which the stack was eminently useful, before delving into its relationship to lower-level constraints. Since then, our understanding of higher-level cognition has advanced (Simon, 1992) and the focus has increasingly turned to the influence of lower-level cognitive constraints (e.g., Byrne & Anderson, 2001; Meyer & Kieras, 1997). In the meantime, however, the notion of a special goal memory has taken on a life of its own, crossing the line from simplifying assumption to theoretical construct. Psychology can ill afford this kind of error, with its potential for propagating through the theoretical food chain. Thus, one of our goals in this article is to put special goal memory to rest as best we can, by offering our own model of cognitive goal representation based on first principles of memory.

3. The goal-activation model: constraints and predictions

For a theoretical foundation on which to build a model of memory for goals, we turn to ACT-R (Anderson & Lebiere, 1998). The ACT family of theories has a long history of integrating and organizing psychological data (e.g., Anderson, 1983; Anderson, Bothell, Lebiere & Matessa, 1998). The current version, ACT-R, derives important constraints from asking what cognitive processes are adaptive given the statistical structure of the environment (Anderson, 1990; Anderson & Milson, 1989). It has also been broadly tested, and offers both algebraic and computational formulations to work with. Finally, ACT-R is, from one perspective, a set of inter-related mechanisms rather than a monolith, and when one of those mechanisms appears to be incorrect, one can test it by “turning it off” and attempting to reduce its functionality to other mechanisms (and make new predictions in the process). This is the approach we take here, by “turning off” ACT-R’s architectural goal stack and working strictly with its ordinary working-memory mechanisms.

We start with the premise that the most active trace in memory is the one that is sampled.

(We use *sample* and *retrieve* interchangeably to refer to the process of bringing information from memory into focal awareness for processing.) Thus, for a particular goal to direct behavior, it must be the most active goal in memory. We develop the implications of this premise below, after which we test them against empirical data from a means-ends task.

3.1. The interference level

In human memory (unlike in silicon memory, which is often a tacit metaphor), old items decay gradually rather than instantaneously, forming a mental clutter that can make it difficult for the system to find the correct item. Interference from this kind of clutter has dominated psychological theories of forgetting, and it plays a critical role here. We will refer to the collective effect of distractor (nontarget) goals simply as the *interference level.*

Informally, the interference level represents mental clutter. Formally, the interference level is the expected (mean) activation of the most active distractor. This definition provides a concise way to assess the likelihood of sampling the correct goal. When the system samples, it will get the most active goal. Whether this is the target or the most active distractor depends on their relative activation values. If the target is *above* the interference level, then it is *more* likely to be sampled than any distractor. If the target is *below* the interference level, then it is *less* likely to be sampled than some distractor. As long as a goal is above the interference level, we say that it *directs behavior.*

An important caveat is that being above the interference level does not guarantee that a goal will be sampled on a given system cycle. In ACT-R, activation is subject to noise—the activation of each item fluctuates about an expected value. Therefore, there is always some probability that the target dips below the activation of some distractor on a given cycle, so the system stands to benefit from maximizing the activation distance by which the target stands above the interference level. Appendix A develops a formal model of the relationship among the interference level, activation noise, and strengthening (defined next), and works through an example.

3.2. The strengthening constraint

Given the interference level, how does the system switch to a new goal? A new goal will suffer proactive interference from old goals, so to direct behavior it has to become more active than the interference level. That much seems obvious, but there is a less obvious consideration, involving the feedforward effect of strengthening an item—the more active an item is now, the higher the interference level later when that item is not the target. Thus, encoding a new goal in memory is more complex than it might seem, in the sense that it is subject to functional constraints. The system must strike a balance between making a new goal strong enough to direct behavior now but not so strong that it interferes excessively later once it is satisfied. These functional constraints have behavioral implications, in that they help predict the amount of time that the system needs to spend encoding a new goal.

Goal activation is determined by the equation below, adapted from the Base Level Learning Equation of ACT-R (Anderson & Lebiere, 1998).[2]

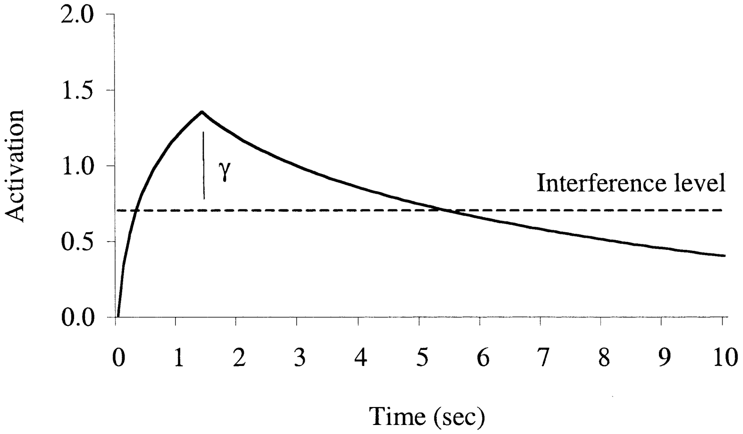

Fig. 1. The time course of activation of a new goal (solid ink) and the interference level due to old goals (dashed ink). The time course is a function of Eq. (1) (see text). Strengthening (on the left) rapidly builds up activation, and is followed by gradual decay (on the right) as the system turns to other processing. The probability of sampling the new goal increases with γ, the amount by which the current goal is more active than the interference level. The interference level is developed formally in Appendix A.

$$m = \ln\left(\frac{n}{\sqrt{T}}\right) \tag{1}$$

m is the activation of the goal (and of memory elements generally, in ACT-R). *n* is how often the goal has been sampled over its lifetime. *T* is the length of the goal's lifetime, from encoding time to present. Thus the equation defines activation as a function of retrieval frequency.

The time course of activation for a goal is shown in Fig. 1, together with the interference level. Time is on the abscissa and goal activation is on the ordinate. The curves show expected activation. The time course has two phases, strengthening and decay, both computed directly from Eq. (1). Initially, on the left of Fig. 1, the goal has just been formulated and is being rapidly strengthened to overcome the interference level. At its peak activation, this goal directs behavior, because it is above the interference level. However, as we elaborate below, this high activation level is impossible to sustain once the system actually begins to engage in task-related operations. So goal activation begins to decline, and eventually drops below the interference level (absent some deliberate intervention to keep it active). This decline is functional in its own right, in that it helps prevent perseveration. Automatic decay of the goal in control ensures periodic re-evaluation of the system's cognitive direction.

It is important to emphasize that Eq. (1) is responsible for both trajectories in Fig. 1—the steep initial increase in activation, and the slower subsequent decay. The two phases differ only in the *rate* at which the goal is sampled. During strengthening, the system devotes all its cycles to sampling, causing a rapid buildup of activation with a rapid increase in *n*. Then, the system turns to other activities, using the goal for direction. Such activities might include formulating yet another goal, or might involve manipulating the task environment in a goal-directed manner. In any case, n now increases at a sharply lower rate while T continues to increase at the same rate. This discontinuity from concentrated strengthening to other task-related activity is abrupt enough that activation of the new goal begins to decline.

Two functional constraints flow from Eq. (1), which together bound goal-encoding time from below and from above. First, strengthening is serial—one sampling event occurs at a time. Thus in Fig. 1 the trajectory during strengthening is sloped rather than being a vertical "step." This is an instance of the relatively uncontroversial assumption that activation increases with exposure to a stimulus (e.g., Stretch & Wixted, 1998). Consequently, strengthening time is bounded from below by the time needed to bring the new goal above the interference threshold. (Appendix B develops the assumption that one sample can occur every 100 ms.) Second, strengthening is cumulative—its effect on n persists throughout the goal's lifetime. Thus although the system could improve its memory for a particular goal without limit by spending more time strengthening, life is not so simple that we have only a single goal. Goals often arise every few seconds, so the interference level is a critical limit on how much activation the system can afford to invest in any particular one. In Fig. 1, the cost of too much strengthening of now-old goals can be seen in terms of γ, the amount by which the new goal at its peak is more active than the interference level. The smaller is γ, the less accurate is memory for the current goal because activation noise is more likely to elevate an old goal above the new goal.

The strengthening process makes testable predictions through its role in *planning*. By planning we mean a process of mental simulation in which the system imagines a sequence of problem states starting with the configuration shown in the task environment. The purpose of planning might be to evaluate a candidate move by playing out its consequences, as in lookahead search (Laird, 1984; Newell, 1990), or to decompose a "large" goal into smaller goals, as in means-ends analysis (Ernst & Newell, 1969; Newell & Simon, 1972). For planning to succeed, each intermediate state (goal) must be immediately and reliably available to the cognitive system, which has to transform that state into its successor by means of task-related cognitive operations. The strengthening process is what makes the state immediately and reliably available, by making that state more active than its competitors. Thus strengthening extends in time, but this time is bounded because the system needs to avoid too much of a good thing.

Planning often incorporates backtracking—returning to an intermediate state in the problem space after pursuing other paths. Planning with backtracking differs from the restricted notion of planning introduced above, in which the system plans each new path by starting over from the external problem configuration. Backtracking raises the question of how the system retrieves an internal state from memory. The goal stack simplifies this problem away, because it preserves old states in the order required for backtracking. In the goal-activation model, retrieval of intermediate states from memory is directed by the priming constraint, as we discuss next.

3.3. The priming constraint

Given the interference level, how does the system resume an *old* goal that was suspended earlier and now needs to be retrieved? A suspended goal suffers retroactive interference—it

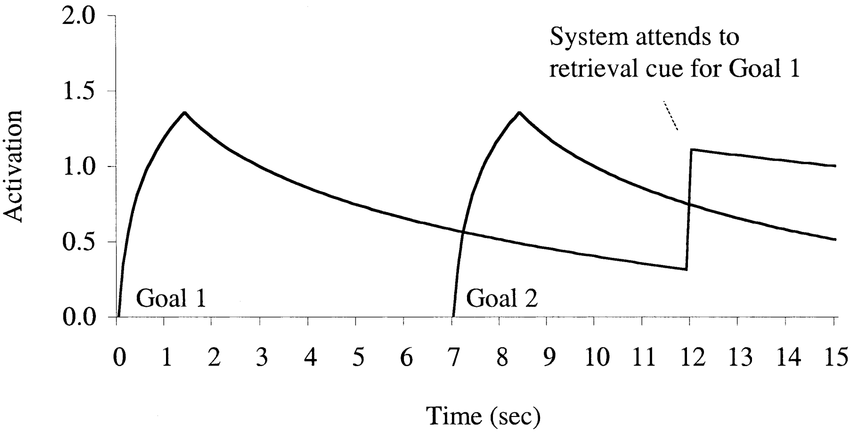

Fig. 2. Effects of associative priming on an old goal. Goal 1 is encoded first, then Goal 2, then a retrieval cue is encountered for Goal 1, making it more active again. Without a retrieval cue, Goal 1 could never become more active than Goal 2 (except possibly through transient activation noise).

will be "buried" or "masked" by its successors in terms of activation—so to direct behavior it has to be made more active than the interference level. Short of a "pop" operation for clearing intervening goals from memory, what could make an old goal the most active again?

Activation can be analyzed in terms of two components, one based on history and one based on context. This is a Bayesian approach, in that the assumption is that the memory system integrates prior experience with current evidence to predict current need for a memory element (Anderson, 1990; Oaksford & Chater, 1998). This predicted need is quantified as the item's activation—active items are ones that history and context suggest are relevant now. In ACT-R's declarative memory system, the effect of history is captured by Eq. (1), and the effect of context is captured by the influence of *cues* in the mental or environmental context. Cues provide associative activation, or priming, to a target item to which they are associatively linked. ACT-R's priming mechanism has its own algebra, which has been used to account for phenomena like the fan effect (Anderson & Reder, 1999) and individual differences in working memory capacity (Lovett, Reder & Lebiere, 1999). For our purposes, priming is important at an abstract level. Priming is the only way to overcome the retroactive interference that affects an old goal. In other words, "Now what was I doing?" can only be answered by generating the right cue as a reminder.

The priming constraint is illustrated in Fig. 2. The first event is strengthening of Goal 1. The second event is strengthening of Goal 2, which results in Goal 2 directing behavior. (The interference level is omitted for clarity.) The third event is the abrupt reactivation of Goal 1 as the system encounters a retrieval cue. This reactivation causes Goal 1 to direct behavior again.

The priming constraint has an important implication for cue availability. Not only must the system attend to the cue when trying to resume a target goal, the cue must be associatively linked to the target goal in the first place. In associative learning mechanisms generally and in ACT-R in particular, links between cue and target are often formed by co-occurrence—the cue and the target have to enter the system's focal awareness at roughly the same time. In

ACT-R, the specific rule for associative learning is that the cue must be in focal awareness when the target is sampled from memory. The implication is that a goal and its retrieval cue must have been sampled together—before the goal was suspended. This means that before the goal was suspended, the cue must have been available in the mental or environmental context.

If a cue must be available before goal suspension, this has implications for the task environment. Specifically, when the task is such that the solution path cannot be memorized, the task environment must contain *means-ends cues* if old goals are to be retrievable. Means-ends cues are defined by two attributes. First, they will be "obvious" in some sense—obvious enough, ideally, that it would be difficult for the system *not* to process them. This obviousness must hold both when the goal is suspended and when it is resumed—it does no good for a cue to be present at retrieval time if no link between the cue and the target has been established. Second, means-ends cues will prime the target but relatively few distractors. This is simply the general principle that cues work better when not "overloaded" (Anderson & Reder, 1999; Roediger & Guynn, 1996). In general, retrieval cues need not be environmental—they can be drawn from long-term knowledge about the task and thus be part of the problem solver's internal mental context. For example, procedural skill can be interpreted as a chain of associative links that reliably cue the next step. However, if the solution path changes randomly from trial to trial, as it does in the Tower of Hanoi puzzle we use to test the model, then knowledge-based cues are not sufficient. That is, if the correct next step is unpredictable, then knowing what it was previously will not help now. Under these circumstances, cues that prime pending goals must in fact come from the environment.

3.4. Summary of constraints and predictions

When the cognitive system asks, "What am I doing?" it is in effect sampling memory for a goal. This sampling is complicated by the interference level, which represents the activation of the most active goal that is not the target. To direct behavior, a new goal must be strengthened to overcome the interference level, and a suspended goal must be primed by a retrieval cue to overcome the interference level. The strengthening and priming constraints make predictions both for behavior and for the structure of the environment. The strengthening constraint predicts that setting a new goal takes time. This time lag should be evident in situations that require looking ahead through mental simulation—as a plan is being formulated, each intermediate step needs to be strengthened to be a reliable input to the operator that transforms it into the next. The priming constraint predicts that cues must be available that the system can encode with a goal and use to prime retrieval later.

4. Memory for goals in the Tower of Hanoi

Means-ends analysis is sufficiently important to human behavior that it might be viewed as the opposable thumb of cognition. It certainly allows us to grasp and manipulate complex concepts piecewise when the whole is too difficult to manage. The Tower of Hanoi puzzle is, in turn, foundational in the study of means-ends analysis (Anzai & Simon, 1979; Egan & Greeno,

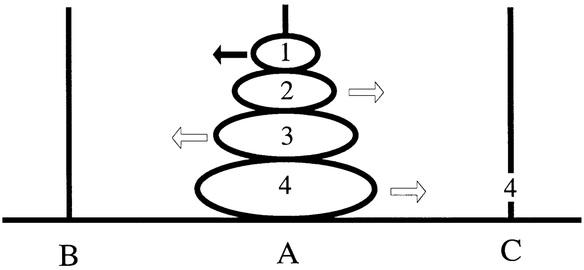

Fig. 3. A problem state in the Tower of Hanoi and an illustration of the goal-recursion algorithm on a planning move. All disks are currently on peg A. Arrows mark goals formulated by the algorithm. Planning begins with the goal to move disk 4 to peg C (4: C) and then formulates subgoals 3: B, 2: C, and 1: B. Planning ends with move 1: B (filled arrow).

1973; Karat, 1982; Kotovsky, Hayes & Simon, 1985; Ruiz & Newell, 1989; Simon, 1975; VanLehn, 1991; Zhang & Norman, 1994). Among activation-based simulations of goal memory (Anderson & Douglass, 2001; Byrne & Bovair, 1997; Goel, Pullara & Grafman, 2001; Just, Carpenter & Hemphill, 1996; Kimberg & Farah, 1993), there are a number that simulate Tower of Hanoi data and that serve to highlight the questions we hope to address here. Anderson and Douglass (2001) dispense with the architectural ACT-R goal stack and instead represent goals as ordinary declarative memory elements. However, the priming constraint is met the traditional way, with a memory representation that maintains a LIFO goal ordering. A subgoal is automatically encoded with a pointer back to its parent goal, and this link identifies the correct target at retrieval time. A retrieval error can occur, but only when the correct goal decays below threshold, not because an incorrect goal intrudes. Thus the model abstracts away from the functional consequences of interference. Goel et al. (2001) also use a stack representation, though a goal can again decay below threshold. Just, Carpenter and Hemphill (1996) appear to use a stack as well, and though goals can decay below threshold, the simulation has a "recovery" mechanism with special access to decayed items that is able to save them if necessary. In our simulation, the strengthening process has no such special access—a decayed goal must receive associative activation from a cue to become retrievable. The goal-activation model thus makes stronger predictions about the existence of cues in the task environment.

Relative to other models, we wish to take seriously the basic forgetting processes of interference and decay and their effects on memory for goals. This appears to require dispensing with special-purpose goal-management mechanisms, whatever their form, and asking what more general strategies are available to the system to remember pending goals.

4.1. A sample problem

The initial state of a typical four-disk Tower of Hanoi problem is shown in Fig. 3, together with the final destination for the largest disk (labeled simply "4"). The arrows in the figure illustrate the *goal-recursion* algorithm (Simon, 1975) deriving the first move from the initial state. The top-level goal is to move 4 from its source peg (A) to its final destination (C). However, this move is blocked because all three smaller disks are in the way—one disk is

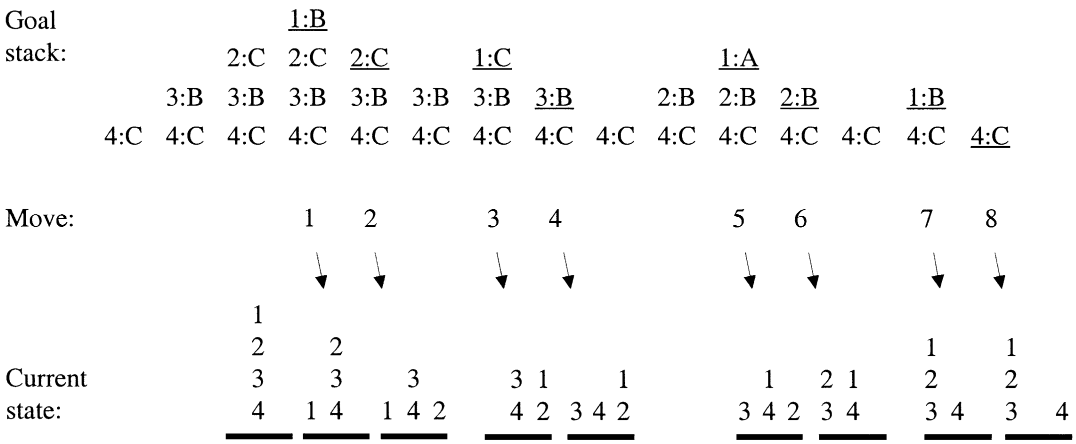

Fig. 4. The first eight moves from the problem state in Fig. 3. Move numbers are shown across the middle. The states before and after each move are shown across the bottom. Goals are shown across the top, in a stack representation. Underlined goals can be achieved immediately, and the rest are pending. For example, starting at the left, the first goal is to move disk 4 to peg C (4: C). This cannot be made immediately so the system formulates 3:B and 2:C, then makes 1:B. After move 8, disk 4 is in position.

said to *block* another if it is smaller than the other and rests on the other's source or destination peg. The goal-recursion algorithm begins with the largest out-of-place disk (the *LOOP* disk) and focuses on each smaller disk in turn, asking where that disk needs to be to unblock the previous disk (the unfilled arrows in Fig. 3). Eventually the algorithm arrives at 1, which can be moved immediately (the filled arrow). After 1 is moved, 2 can be moved, but then 3 is blocked, so the algorithm must be invoked again. In this example we have purposely omitted final destinations for disks other than 4 to make a distinction between final and intermediate destinations. Smaller disks have a greater number of intermediate destinations than larger disks, because they have to move more often to unblock larger disks. Thus 1 has a new intermediate destination on every second move, whereas the largest disk (4 in this example) moves only once; after it has been moved, it can be completely ignored for the rest of the trial.

A *goal* in this context is an association between a disk D and a destination peg P. We designate a goal (and a move, where appropriate) as D:P. The first goal produced by the goal-recursion algorithm is 4:C, the second 3:B, and the third 2:C. The first of these (4:C) is readily inferred from the display by comparing the current state of the puzzle to the final state. However, the recursive subgoals formulated in service of 4:C are not perceptually available. These must be replanned after every move using the goal-recursion algorithm or they must be stored in memory and retrieved at the right time. For example, once 1:B has been made, the next move could be planned by focusing on 4 again, then on 3, and then on 2. Alternatively, if a goal for 2:C had been stored in memory during the first pass of the algorithm, and if that goal could be retrieved now, then this second planning episode could be spared; memory would indicate 2:C.

The initial moves from the start state shown in Fig. 3 are shown in Fig. 4. This eight-move sequence is the optimal sequence for unblocking disk 4 and moving it to its destination. A stack-based representation of pending goals is shown across the top of Fig. 4 (we will refer to this later as we examine how an existing, stack-based model accounts for empirical data on this task). Each of the eight moves is numbered across the middle of Fig. 4, and the state of the puzzle before the first move and after each move is shown across the bottom. From the initial state, the system encodes goals 4:C, 3:B, and 2:C, then finally makes move 1:B (move 1). The system then makes move 2:C, because disk 2 is clear. Goals that can be immediately achieved (because their disks are clear) are underlined.

Fig. 4 illustrates the benefits that would accrue to a system endowed with a goal stack. The top of the figure shows a stack-based organization of goals. A pending goal is always at the top of the stack when it is needed. For example, after move 2 (2:C), the top of the stack indicates the correct next move, which is 3:B. If this move could be made at this point, the system would have the information it needed to make it. If the move is blocked, as it is in Fig. 4, the system has the information it needed to infer the necessary subgoals. Either way, the goal stack makes the right information available at the right time (though this is not true for all problem solving tasks; VanLehn et al., 1989). To do so it makes problematic assumptions about the persistence of memory. For example, goal 3:B was encoded (and pushed on the stack) before move 1, but only retrieved (by popping 2:C) after move 2. This interval spans five to 10 s empirically, and is filled with distracting activity in the form of other mental operations related to the task but not to that particular goal. Normally one would expect access to a trace to be lost under such circumstances. The stack also preserves goal order without any of the positional confusions typical of human memory (e.g., Nairne, 1992). In sum, for all its usefulness in representing task constraints, the goal stack fails to represent cognitive constraints.

4.2. A goal-activation simulation

We tested the goal-activation model by constructing a simulation in ACT-R's production system and fitting it to and testing it against an existing Tower of Hanoi data set. The simulation implements the goal-recursion algorithm and memory processes that encode and retrieve goals. Goal encoding occurs during goal recursion and involves strengthening each goal as it is formulated. Retrieval occurs after the model makes a move and needs to decide what to do next, and involves selecting a retrieval cue from the task environment and then trying to retrieve the associated goal. Below we describe these processes and how they realize the strengthening and priming constraints. A more detailed description, with pseudoproductions and control-flow graphs, is given in Appendix B.

4.2.1. The strengthening constraint: implications for planning moves

The hypothesis represented in the simulation is that strengthening each goal is critical during *planning moves*—those on which the goal recursion algorithm runs. Strengthening during planning is important for two reasons. First, each goal is the basis for formulating its subgoal, and although the initial goal on such moves can be formulated directly from perceptual information, its subgoals are intermediate mental products and have only mental representations. Because a subgoal resides in the head only, it must be active enough to be a reliable input to the next level of recursion. The cost associated with an insufficiently active subgoal is that a memory failure breaks the chain of inference and forces the system to start over. The second reason to strengthen goals is to support their retrieval later. The encoded goals represent a plan, and it pays to be able to retrieve the steps of this plan (as we demonstrate later). Retrieving a plan step requires not only a retrieval cue, but also depends on the strength of other goals primed by that cue. For example, a cue might prime goals from previous trials, and those could interfere if they have more base-level activation than the target (per Eq. (1)).

For these reasons we would expect strengthening to be essentially automatic on moves that require planning. Latency on such moves should therefore accommodate adequate strengthening time for each goal, but also time to actually use each goal to formulate the next (Anderson & Douglass, 2001). As we show below, the simulation suggests that both processes can be accommodated in the time available, and in Appendix A we generate a theory-based estimate of how much time is allocated to each.

4.2.2. The priming constraint: implications for cue selection

The priming constraint says that retrieval cues are necessary to resume an old goal because the old goal is affected by retroactive interference from intervening goals and needs the activation boost. The prediction for the Tower of Hanoi is that for every goal retrieved there is a means-ends cue—an object likely to be attended both when the goal is suspended and when it is to be retrieved, and that primes that goal but no (or few) others. The notion of cues of this kind does not appear to be documented in the literature on the Tower of Hanoi, even in research focused on perceptual problem-solving strategies (Kotovsky et al., 1985; Ruiz & Newell, 1989; Simon, 1975; Zhang & Norman, 1994). If we can identify such cues, and demonstrate their sufficiency with a simulation, this would contribute to general understanding of the domain. It would also set the model up for experimental tests involving manipulations of cue availability.

A goal in the Tower of Hanoi is to move a disk to a peg, so a logical candidate for a cue is the disk itself. At goal-encoding time the system has to process the disk anyway, in the course of applying the goal recursion algorithm to decide where the disk should go. Thus, the step of linking the disk associatively to its target peg is actually hard to avoid. At retrieval time, which comes immediately after the system moves a disk (and has to look for another disk to move), it makes sense to ask what other disk might now be movable. Indeed, the purpose of most moves in the puzzle is to free up other disks. The only question is which disk to focus on, but here there is also considerable guidance, both from the solution strategy and perceptually from the puzzle display. The solution strategy is the goal-recursion algorithm, and if one simply inverts (or “unwinds”) it once a disk is moved, the algorithm specifies which disk to try to move next. The next disk to try is typically the one uncovered by the move just made (though there are exceptions; the complete set of cue-selection heuristics is described in Appendix B). The uncovered disk is again hard to miss, due to its spatial proximity to the disk just moved.

Overall, then, there is high face validity to the notion that disks themselves function as means-ends cues. This makes a clear prediction for the Tower of Hanoi: The presence or absence, or relative availability, of disks as cues should affect performance by affecting the ability to retrieve goals. This prediction will be important to test experimentally; here we take

Table 1
Move Latencies (sec) from the Tower of Hanoi

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
4-disk	9.7	2.5	2.8	2.2	3.1	2.0	2.4	2.0	6.6	2.4	2.9	2.1	5.4	2.2	2.0
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
5-disk	14.92	2.27	2.68	2.42	3.39	2.03	2.67	2.40	4.92	2.07	2.68	1.83	2.50	1.80	2.14
	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
	1.79	8.95	2.46	2.76	2.24	3.00	1.89	2.28	2.13	5.22	2.41	2.81	2.24	4.56	2.18

Note: Move number appears above each latency. Four-disk data are from Anderson and Lebiere (1998, Fig. 2.7) and are available at http://act.psy.cmu.edu. Five-disk data are from Anderson, Kushmerick, and Lebiere (1993, Table 6.3), and omit move 31.

the initial step of demonstrating the sufficiency of means-ends cues to provide access to pending goals.

4.3. Testing the simulation

To test the simulation, we first fit it to latency data to show that memory for goals explains substantial variance in the simulation's behavior. We then froze the code and parameters and conducted a zero-parameter test of the model's predictions for error rates.

The data we examine are from a study reported in Anderson et al. (1993) and Anderson and Lebiere (1998). This study is attractive in part because participants were guided toward use of the goal-recursion algorithm.[3] This is important because our task analysis assumes use of this algorithm. The study is also attractive because it took steps (by using a variety of task configurations, and not just tower-to-tower ones) to minimize the possibility that participants would memorize move sequences. This is important because it means that participants were forced to use means-ends cues to retrieve old goals, rather than using internal (procedural) cues learned from previous trials. The task environment for the experiment was a computer-based simulation of the Tower of Hanoi, in which participants made moves using the mouse. Several different measures were included in the original report (Anderson et al., 1993), but the two measures of interest here are latencies on perfect trials and number of stray moves overall. Latencies are simply the time between one move and the next. A "perfect" trial is one on which the solution was achieved in the minimum number of moves. The chances are quite good that on such trials, participants were using the strategy they were taught (Anderson et al., 1993). We then test predictions of the model against an independent measure—the number of stray moves on trials not included in the response-time measure. The simulation makes a stray move (or a sequence of stray moves) if it happens to retrieve the wrong goal from memory (due to noise in activation levels). If we can predict error rates based on the simulation's fit to latencies, this would be a strong test of the goal-activation model as a whole.

4.3.1. Fitting the latency data

Empirical latencies are presented in Table 1 and plotted against simulated latencies in Fig. 5 (four-disk problems) and Fig. 6 (five-disk problems). The simulation accounts for 99% of

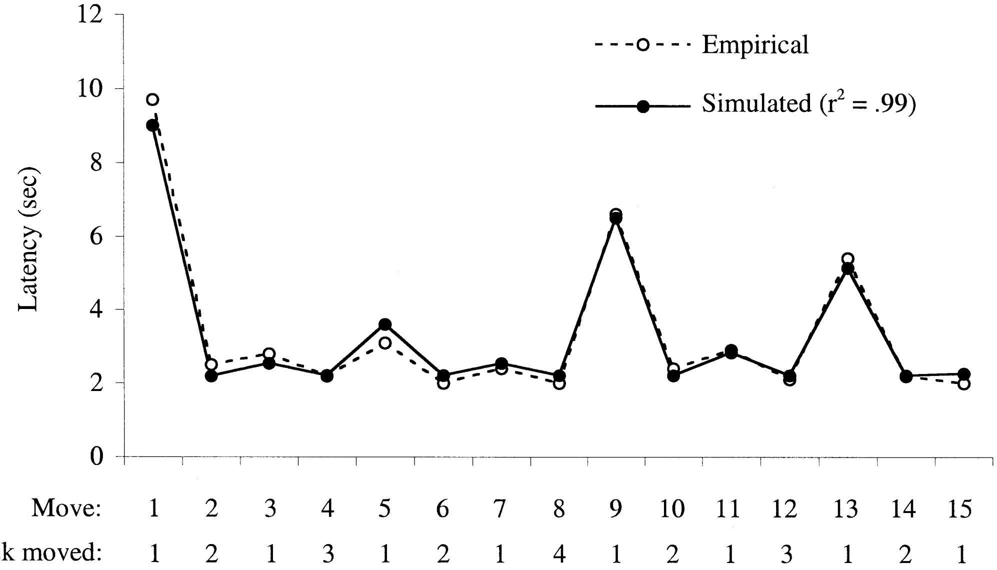

Fig. 5. Move latencies for four-disk problems. Empirical values are from Table 1. Simulated values are from the goal-activation model.

the variance in four-disk latencies and 95% of the variance in five-disk latencies. (A stack-based simulation fits the latency data equally well, a point we return to later.) Below we examine the patterns in the four-disk data (Fig. 5), showing how the mechanisms we described above account for each. The same explanations apply to the five-disk data.

The first pattern consists of the large peaks at moves 1, 9, and 13. In the simulation, these peaks are due to the strengthening process, which is triggered when it is time to plan to unblock the LOOP disk. Consistent with the strengthening constraint, the model generates a plan and encodes each step in memory. Strengthening each goal in a given plan takes one to two seconds, adding up to the height of the large peaks (after the baseline move time of about two seconds is subtracted).

The second pattern is the decrease in height of these large peaks within a trial. That is, move 1 takes the longest, move 9 is faster, and move 13 is faster still. In the simulation, the peaks shrink because there is one fewer step to each successive plan—the LOOP disks from previous plans can be ignored, because they have reached their final destinations. Thus at move 1 the model encodes four goals (for disks 4, 3, 2, and 1), but at move 9 it encodes only three (for disks 3, 2, and 1).

The third pattern is that even-numbered moves are all equally fast, taking just over 2 s. In the simulation, the correct move at even-numbered moves is indicated immediately by the Don't-undo heuristic (described in Appendix B). This heuristic is based on general considerations of cognitive and movement efficiency, and is quickly acquired without any specific instruction (Karat, 1982; VanLehn, 1991). Essentially, even-numbered moves are fully determined by task constraints. Latency on these moves thus primarily reflects time for perceptual and motor actions.

The fourth pattern consists of the small peaks at moves 3, 7, and 11. In the simulation,

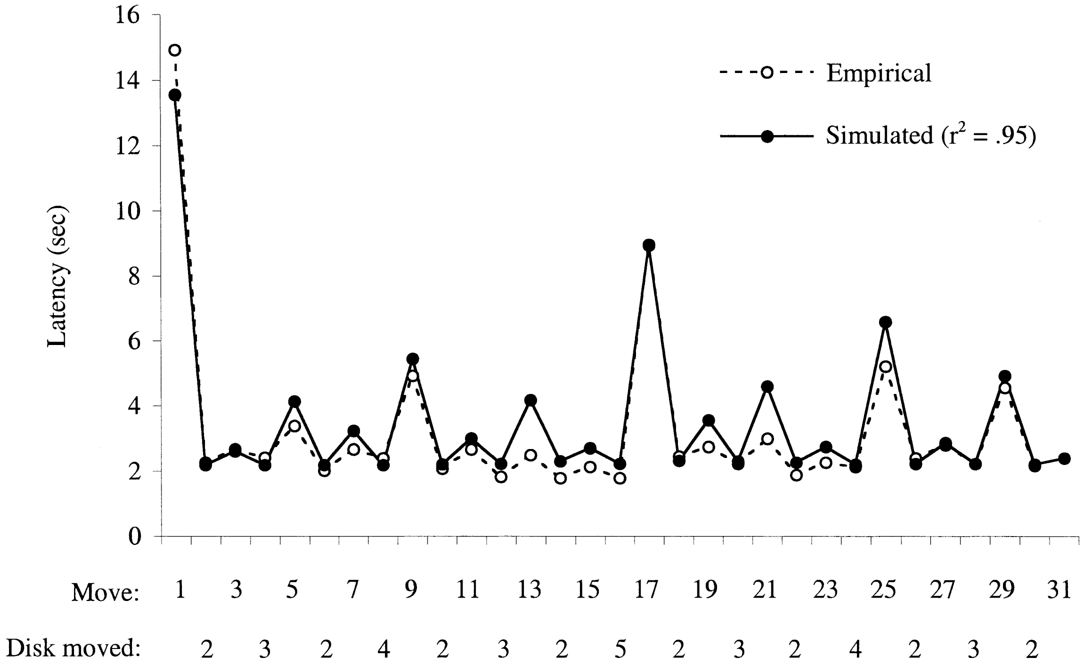

Fig. 6. Move latencies for five-disk problems. Empirical values are from Table 1. Simulated values are from the goal-activation model.

these peaks reflect time to select the cue and retrieve a goal. They are smaller than they might be, because the simulation does not always attempt retrieval; sometimes it uses a domain-specific heuristic that we refer to as One-follows-two (but which is also known as the move-pattern strategy; Karat, 1982; Simon, 1975). This heuristic is to follow a move of disk 2 by placing disk 1 on top of disk 2. Like Don’t-undo, it is acquired quickly and without explicit guidance (Karat, 1982; VanLehn, 1991). However, because it is domain-specific, rather than general like Don’t-undo, we assume that it applies only stochastically (Karat, 1982, also assumed that Don’t-undo always applied and that One-follows-two was stochastic). In our simulation, the probability that One-follows-two applies when it can is 0.5.

The fifth and final pattern is the medium peak at move 5 (Fig. 5). In the simulation, medium peaks reflect a combination of factors. First, the Don’t-undo and One-follows-two heuristics do not apply, so the simulation always falls back on goal retrieval. Second, the to-be-moved disk is blocked, so the simulation must invoke the goal-recursion algorithm to clear it. Third, the to-be-moved disk is larger on this move than on small-peak moves, so the goal-recursion algorithm runs longer.

4.3.2. Latencies from a lesioned model

Having shown that the simulation is sufficient to account for the latency data, it’s important to ask whether memory for goals is a necessary part of this account. It could be, for example, that the various move-selection heuristics described above suffice to reproduce

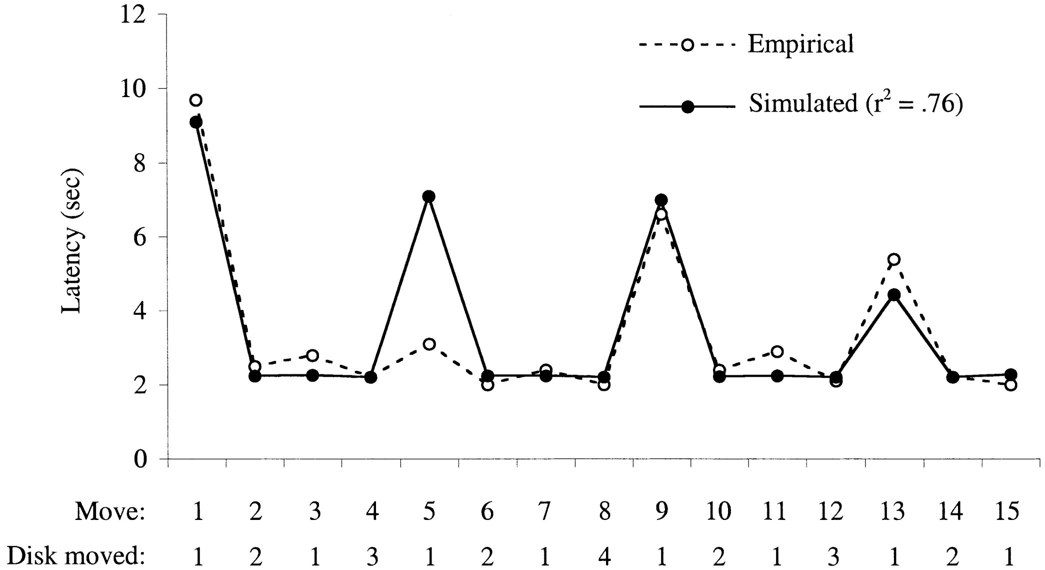

Fig. 7. Move latencies for four-disk problems. Empirical values are from Table 1. Simulated values are from the goal-activation model lesioned to disable goal retrieval.

the latency patterns, and that goal retrieval explains little or no variance. In this case the simulation would not be much of a test of the goal-activation model.

To assess how much variance is explained by memory for goals, we lesioned the model by disabling the goal retrieval process. With this process disabled, the simulation must always replan starting from the LOOP disk on moves where no move-selection heuristics apply. The new latency predictions are shown in Fig. 7 and Fig. 8. The four-disk predictions (Fig. 7) remain reasonably close—only the latency for move 5 is noticeably overpredicted. However, the five-disk predictions show the role of memory for goals more clearly. The lesioned simulation substantially overpredicts latencies on at least four moves (5, 9, 13, and 21), and now accounts for only 52% of the variance.

This lesioning exercise makes two main points. First, it demonstrates that the model's memory for goals is functional in that it accounts for important behavioral variance. Second, it demonstrates that the means-ends cues and cue-selection heuristics we identified are sufficient to access the model's memory for goals. If we have an accurate memory model, then such cues and heuristics should work for people as well.

4.3.3. Predicting the error data

Having fit the simulation to the latency data, we can now test it against the error data. Our model predicts that errors are caused by activation noise, which can make the wrong goal strongest and send the simulation down the wrong path. For example, in the problem in Fig. 4, the correct goal to retrieve after move 2 is 3:B (we know this because after move 2, 3:B is on top of the stack at the top of the figure). Retrieving 3:C instead, say from a previous trial, would direct the simulation to make 1:B next instead of 1:C. Thus, the more the

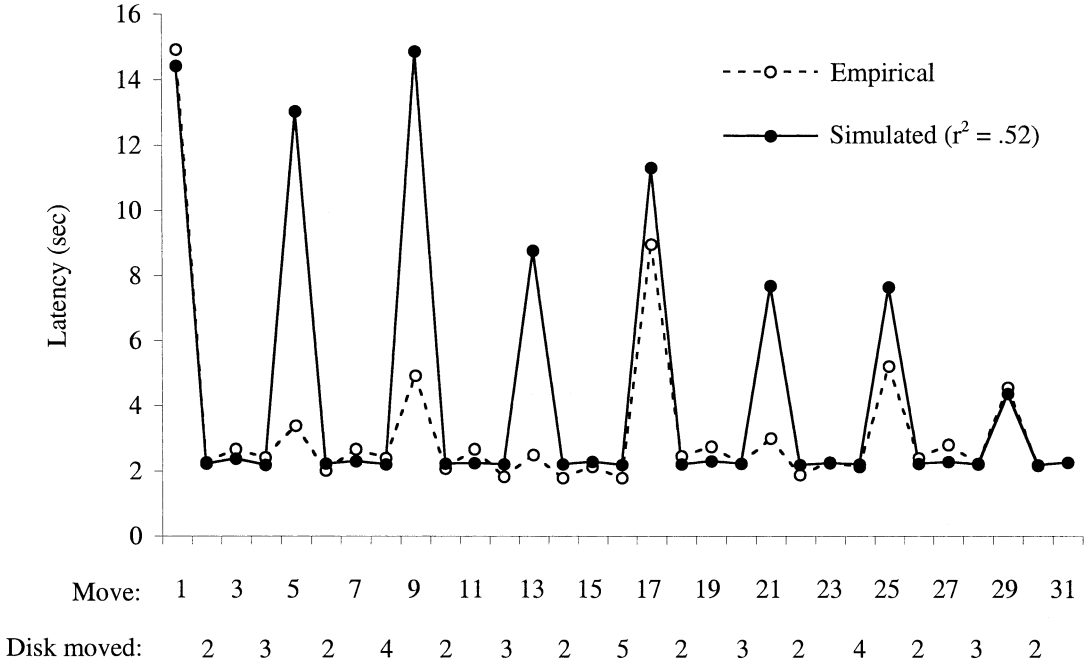

Fig. 8. Move latencies for five-disk problems. Empirical values are from Table 1. Simulated values are from the goal-activation model lesioned to disable goal retrieval.

simulation relies on memory for goals, the more opportunities it will have to retrieve the wrong goal and stray off the optimal path. The prediction for human behavior, for which there is independent empirical evidence (Goel et al., 2001), is that errors should increase with problem size, as larger problems involve greater reliance on memory.

To test this prediction against the current data set, we can compare error rates across four- and five-disk problems. From the lesioned simulation (Fig. 7 and Fig. 8), we know that the intact simulation relies on memory more during five-disk problems than four-disk problems. Consequently, the simulation should make many more errors on five-disk problems.

Fig. 9 shows predicted, observed, and optimal path lengths for four-disk and five-disk problems. The simulation predicts average path lengths of 17.7 moves for four-disk problems and 50.6 moves for five-disk problems (based on Monte Carlo simulations of 2000 trials). Optimal path lengths are 15 and 31, so as a proportion of optimal path length, predicted errors are 18% for four-disk problems and 63% for five-disk problems. Thus the simulation does indeed make more errors on five-disk problems, in relative as well as absolute terms. As importantly, so did participants. Observed path lengths are 19.2 and 53.5 (Anderson et al., 1993), both within a few moves of the predicted number.

There are several points to emphasize about this test of the simulation. First, the predicted error pattern is relatively complex, involving a marked effect of problem size rather than just a simple proportion of optimal path length. Second, the prediction is zero-parameter—the

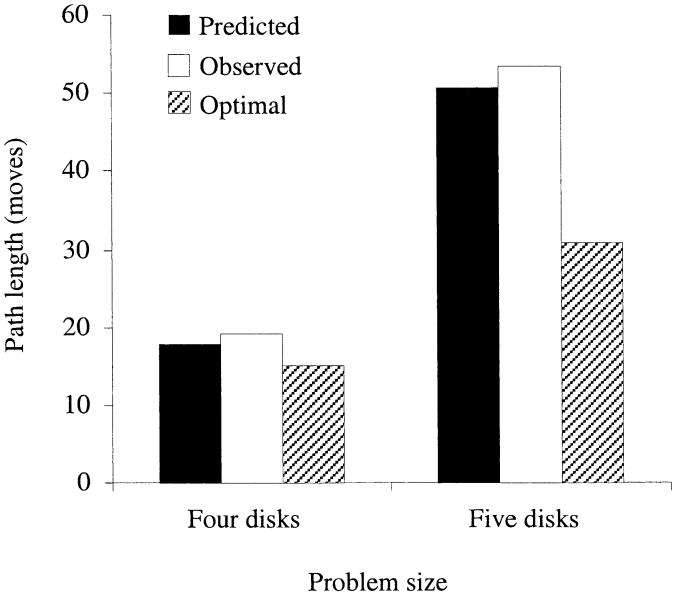

Fig. 9. Average path lengths for four- and five-disk trials. Predicted values are from the goal-activation model and observed values are from Anderson et al. (1993). The optimal path is one on which no stray moves are made.

model was fit to the latency data, then frozen to test against the error data. Finally, the predictions are not just qualitatively accurate, but quantitatively accurate—both are within 10% of observed. This test provides strong evidence for the simulation and for the goal-activation model as a whole.

4.4. A stack-based simulation

The Tower of Hanoi has a long history in studies of higher-level cognition, so it's not surprising that even recent models retain some of the assumptions of early cognitive theory. One example is a model by Anderson and Lebiere (1998). In this "AL" model, a push operation adds a new goal to the stack, and a pop operation resumes a pending goal. Like our simulation, this model is implemented using the ACT-R production system, but there are substantial differences in representation that flow from its use of ACT-R's architectural goal stack. One limitation of the AL model is that it says nothing about stray moves. Unlike the ACT-R memory system proper, the architectural goal stack incorporates no theory of error, so the model completes every trial in the optimal number of moves. However, the AL model does fit the latency data closely, and it is useful to examine its explanatory constructs.

The AL model accounts for patterns one and two (the large response-time peaks and their decreasing height, respectively) with a process similar to ours in that it processes each out-of-place disk in the current problem state. However, what this process actually computes is not specified. Though nominally an encoding process, it is a nonfunctioning "stub" with no output or other kind of effect on the model's representations. The duration of the process

(560 ms per disk) is estimated from the data, with no theoretical or functional constraint. Patterns three, four, and five are explained by stack operations. On even-numbered moves, the top disk on the stack can be moved immediately, so these moves take the least amount of time. On moves with small peaks, the top disk is blocked by one disk, so the goal-recursion algorithm has to run one cycle, pushing the blocked disk onto the goal stack. This push operation takes a single production firing, accounting for the small increase in response time. On moves with medium peaks, the top disk is blocked by more than one disk, so the goal-recursion algorithm has to run multiple cycles. Fig. 4, which shows the first eight moves of a four-disk Tower of Hanoi problem, also shows the basic workings of the goal stack, across the top of the figure. Indeed, the goal stack is so concise a representation for tasks like the Tower of Hanoi that its history is less of a puzzle. A tool this useful is hard to abandon, even if keeping it around as a programming convenience in systems like ACT-R means that sometimes it is misinterpreted as cognitive theory.

In comparing the two simulations, two contrasts are worth noting. First, the models differ substantially in their level of detail, particularly with respect to the processes underlying the most marked patterns in the data. In our model, the processes responsible for the large response-time peaks (patterns one and two) are implemented in terms of lower-level mechanisms (see Appendix B) and hence afford detailed scrutiny. In addition, the underlying mechanisms are not ad hoc but existed in the ACT-R production system already, which is important for parsimony. In the AL model, the corresponding processes are unimplemented and hence difficult to evaluate. Second, the models differ in their degree of integration. In our model, the strengthening process that accounts for patterns one and two is functional in that its outputs are inputs to other stages of information processing. These other stages, in turn, account for the other patterns in the data. Such integration is important because it provides additional constraint—processes are shaped not only by the data they must explain but by the functional needs of other processes. In the AL model, in contrast, the process that accounts for patterns one and two is independent of the processes that account for patterns three, four, and five. This independence is a lost opportunity to exploit mutual constraint among processes (Newell, 1973, 1990).

4.5. Summary of the goal-activation simulation

We have described a computational simulation of human performance on a task that requires goals to be suspended and resumed. The simulation achieves a close fit to latency data, demonstrating the sufficiency of a strengthening account of long lags on planning moves, and the sufficiency of means-ends cues for goal retrieval. The simulation also makes predictions about errors based on noisy goal retrieval—predictions that are reasonably complex and remarkably accurate. Finally, the model contributes to our understanding of the Tower of Hanoi puzzle. Despite decades of research, the role of environmental cues (or “affordances”; Hutchins, Hollan, & Norman, 1985) in supporting cognitive operations has not been investigated at this level of detail—perhaps because tacit assumptions about special goal memory meant there was no need to.

5. General discussion

The goal-activation model uses the construct of activation to advance our understanding of goal-directed cognition. The most active goal in memory is the one that directs behavior, because that is what the system samples when it seeks guidance from memory. During planning, as the system decomposes a large goal into smaller goals that are easier to achieve, the goal being decomposed must be highly active in order to be the reference point for the decomposition process. Later, during execution, the same goal may have to be activated again once its subgoals have been achieved. At this point, priming from cues is necessary to overcome retroactive interference from other, newer goals in memory.

The importance of the task environment in problem solving was a parenthetical remark in early cognitive theory (Newell & Simon, 1972), then gained prominence with the display-based and situated-action approaches (Larkin, 1989; Suchman, 1987). Soon after, an analysis of computational requirements made the case that environmental cues are necessary and sufficient for goal reconstruction in dynamic environments (VanLehn & Ball, 1991). More recently, diverse cognitive simulations have indicated the strong dependence of skilled or expert behavior on environmental cues (Altmann & John, 1999; Howes & Young, 1996; Rieman, Young & Howes, 1996). The present model follows the trajectory of this work, generalizing the importance of cues to task environments in which the solution changes from trial to trial and cannot be memorized by rote or recognized immediately.

The model makes some specific predictions and raises some general questions. One prediction that seems readily testable concerns the role of cues in goal retrieval. It should be possible to affect resumption of an interrupted task by manipulating cue availability, not just at goal retrieval time but also at goal encoding time. More generally, as goal states become more complex and cue-selection heuristics less obvious, we would expect problem solvers to plan from the external state more often, as the difficulty of retrieving intermediate states from memory increases. Such planning from scratch, sometimes known as “progressive deepening,” is what our lesioned simulation fell back on because it could not retrieve its goals. Indeed, progressive deepening is quite common in conceptual, ill-structured domains (Jeffries, Turner, Polson & Atwood, 1981; Kant & Newell, 1984; Steier & Kant, 1985). With respect to the strengthening constraint, the algebraic form of the model (relating strengthening, interference, and activation noise) seems to have potential to make quantitative predictions about memory accuracy as a function of rate of change of the environment. It also seems to bear directly on the question of why “switch cost” is so persistent when people are asked to switch often between simple cognitive tasks (Altmann & Gray, in press, 2000; see Monsell & Driver, 2000, for a snapshot of the task-switching literature). Exploring the potential of the strengthening constraint will require both further mathematical development and empirical testing. Finally, the Tower of Hanoi simulation deploys strategies and heuristics, but does not learn them. Human performance on such puzzles evolves in systematic ways (Gunzelmann & Anderson, 2001; Kotovsky et al., 1985; VanLehn, 1991) and varies with emphasis on speed versus accuracy (Fum & Del Missier, 2001). An important step will be to ask how memory strategies themselves evolve, and how they interact with other learning and strategy variables (Fum & Del Missier, 2001).

In our historical review of theorizing about goals, we touched on two instances in the literature in which empirical data were linked with the notion of a special goal memory. With our model in hand as an alternative, we can ask how it would account for these findings. If the goal-activation model is correct, it should provide a plausible account of whatever patterns in the data would otherwise be explained by a special goal memory. The next two sections address these effects, and the two following sections address practical applications of the model—to procedural slips like postcompletion error, and to the effects of commonplace interruptions.

5.1. Inhibition of return to old goals

One empirical finding that has been linked to a special goal memory is the inhibition of return to recent goals (Mayr & Keele, 2000). This is a puzzling effect from the perspective of the goal-activation model, which predicts that recent goals should be easier to return to because they have decayed less. However, this is also a puzzling effect when viewed from a broader perspective—the goal-activation model is not alone in proposing that recency is an advantage in memory. And yet, despite the advantage that recency generally confers on memory for items, inhibition of return is found in diverse cognitive behaviors (Dagenbach & Carr, 1994), from task switching (Mayr & Keele, 2000) to visual attention (Posner & Cohen, 1984) to language processing (Arbuthnott, 1996) to implicit sequence learning (Boyer, Destrebecqz & Cleeremans, 1998). Thus the contradiction between recency effects on one hand and inhibition of return on the other plays out on a much broader scale than memory for goals.

For our purposes, the important point is that the goal stack is no better than the goal-activation model in explaining cognitive inhibition of return. There is no reason that a goal popped recently should be more difficult to push again than one popped a while ago. Indeed, in the current version of ACT-R, decay affects a goal as it affects any other element, once the goal has been popped off the stack. As far as we can tell, the notion that goals have to be inhibited to overcome their special strength (Mayr & Keele, 2000) is an elaboration of the original ACT* mechanisms, rather than a direct implication.

5.2. The intention superiority effect

A second empirical finding that has been interpreted in terms of a special goal memory is the intention-superiority effect. The theoretical claim is that an intention to perform an action (an *action intention*) is stored with greater activation in memory than an intention to observe another actor doing the action (an *observe intention*). Goschke and Kuhl (1993) refer to this claim as the “persistence hypothesis.” This higher activation is assumed to facilitate recognition, and also lexical decision (Marsh et al., 1998), for items related to the action intention. Measures of intention superiority seem to support the notion that memory for something is indeed better the more “goal-like” that thing is. Here we attempt to address this effect within a framework that does not acknowledge a structural difference between memory for goals and memory for other facts and events.

The key observation from our perspective is that the experimental procedure of Goschke and Kuhl (1993) leaves enough slack time for the execution of the kinds of strengthening processes that we assume in the goal-activation model. Specifically, participants had the opportunity to strengthen their action intention for two seconds between the time the action was identified to them and the time that the intention-superiority measure was applied.[4] Two seconds may not seem like much time, but the goals themselves were described in phrases short enough (e.g., "spread the table cloth") to be rehearsed at least once in the time available (Baddeley, 1986).

The cognitive system has several good reasons to take advantage of any opportunity to strengthen an action intention over an observe intention, and particularly over a neutral intention (one associated with no future event). Acting out a script involves recalling its steps, but monitoring or observation involves recognition, which is typically more accurate to begin with. Extra rehearsal of the action would strengthen its individual propositions, improving recall while performing the action script, and therefore incidentally improving the intention superiority measure. Thus, intention superiority could simply reflect strategic memory processing taking place in the temporal chinks of the procedure. Whether such processing would add enough activation to explain intention superiority would best be demonstrated through cognitive simulation. However, our model, in both its algebraic and computational forms, shows quite clearly that a second or two of strengthening can have substantial effects on memory-based performance. Moreover, our model has the benefit of making a prediction: If even this small opportunity for rehearsal is minimized, intention superiority should diminish.

Thus, our analysis suggests that the construct of a special goal memory adds little to our understanding of inhibition of return and intention superiority. If recent goals are difficult to return to, neither a special goal memory nor the goal-activation model (in its current form) has an explanation. And, if goals are remembered better than other memory elements, it simply redescribes the effect to say that they are stored in memory in a special state. A more fruitful approach is to address intention superiority in terms of underlying memory processes. At this next level down, small but critical windows of opportunity for strategic encoding easily explain the effect.

5.3. Postcompletion error

The need to resume suspended goals is surely more general than the Tower of Hanoi, and in this and the next section we apply the goal activation model conceptually to two everyday domains and ask what analytical leverage it may offer. The first domain is postcompletion error (Byrne & Bovair, 1997). This is a memory-based error made while "wrapping up" some common procedural activity. An example might be to take the copies but forget the originals, to take the cash but forget one's card in the teller machine, or to drive away after filling the tank and leave the gas cap sitting on the pump. In each case, the forgotten action would normally occur after the main goal of the activity is accomplished (making copies, getting cash, getting gas). People usually remember such actions, but do forget them more often than chance (Byrne & Bovair, 1997). This is an important pattern to address, not just because it appears to be mediated by goal structures, but because postcompletion actions themselves are goals in the sense that one "knows" to carry them out as part of one's procedural knowledge. Postcompletion error also lies at the intersection of two broader areas of goal-related research. One is the study of goal-neglect (Duncan, Emslie & Williams, 1996), in which patients with frontal lobe damage dissociate declarative understanding of the task from the ability to perform the task (they can talk about it, but they can't seem to do it). The other area is the study of prospective memory (Brandimonte, Einstein & McDaniel, 1996), in which the research questions are often applied and focus on memory for procedures or errands to be carried out at some point in the future after intervening activity.

Byrne and Bovair (1997) modeled postcompletion error using the CAPS cognitive architecture (Just & Carpenter, 1992). Despite a different underlying theory and a different behavioral focus, there is substantial overlap between their model and ours, which we take as converging evidence for both. In particular, an important premise of their model is that the user's procedural knowledge includes an associative link from the main goal to the actions necessary to achieve it. This link spreads activation to the actions as long as the main goal itself is active. For example, the main goal of getting gas primes the action of replacing the gas cap and thereby keeps the action available as a pending subgoal—as long as the tank is not yet filled. In terms of our model, the main goal is the cue that meets the priming constraint. The cues we discussed in context of the Tower of Hanoi reside in the task environment, but the model is agnostic about whether a cue is an attended object external to the system or an attended idea internal to the system. To meet the priming constraint, a cue need only be associatively linked to the target and be attended or active in memory at retrieval time.

The goal-activation model addresses a limitation of the Byrne and Bovair (1997) model, which is the assumption that correct performance depends on the task taking up most of the available working memory capacity. This assumption is implied by CAPS, in which activation moves around among memory elements but the overall quantity is conserved. This conservation means that unneeded (interfering) memory elements lose activation only when other elements gain it. In the Byrne and Bovair model, this conservation causes satisfied goals to interfere with unsatisfied ones if the activation of satisfied goals is not "absorbed" by other memory elements. In other words, insufficient memory load actually raises the interference level. This makes the awkward prediction that errors should increase monotonically as memory load decreases, all else being equal. The theoretical inference we make is that interference cannot be the only kind of forgetting—interfering elements themselves must decay if memory is to serve its function (this argument is tested experimentally by Altmann & Gray, in press). In the goal activation model, decay is indexed by time, not memory load, so a task with a lower memory load (fewer memory elements encoded per unit time) will indeed suffer less interference.

The decay process in the goal-activation model has a cost as well, which is that suspended goals are forgotten gradually, making them harder to resume. With respect to postcompletion error, the model implies that the default tendency is to make such errors, not avoid them. And yet, people seem to remember postcompletion actions successfully most of the time. We take this as evidence of deliberate cognitive operations undertaken to meet the priming constraint—to ensure the existence of an associative link to the postcompletion action, and to ensure attention to the right cue at the right time. One such cognitive operation might be rote associative (procedural) learning—through temporal co-occurrence, the step that precedes a postcompletion action will eventually come to serve as a cue for the action itself. The general implication is that people are able to retrieve suspended goals successfully if and only if there are cues that meet the priming constraint. This is a hypothesis to be tested empirically which may then turn out to be a useful constraint on design and training in situations in which reliable goal resumption is important.

5.4. Cognitive effects of interruption

In our everyday lives, interruptions are continual—the phone rings as you are deeply engaged, some "software assistant" seizes the keyboard unexpectedly as you are planning a document, a message arrives from a supervisor or teammate drawing your attention to something urgent. Each case easily creates a situation in which a goal must be suspended before it is completed, and then resumed later. One senses that interruptions are disruptive, but how disruptive are they, and how (if at all) do people compensate?

Data on how the cognitive system resumes goals after an interruption are sparse and often contradictory. Early on, the work of Ziegarnik (1927) suggested that interrupted goals were remembered better than goals allowed to run to completion, though this effect has been difficult to replicate (Patalano & Seifert, 1994). An important methodological problem in studying the effects of interruption is that experimental control over the timing of an interruption relative to performance of the primary task has been difficult to achieve. As a consequence, operators are often able to adopt compensatory strategies and "work around" the interruption, masking its effects (Latorella, 1996; Zijlstra, Roe, Leonora & Krediet, 1999). Strategic adaptation could well explain the weak existing evidence that participant control over interruption timing has no effect (Gillie & Broadbent, 1989). Indeed, in some circumstances participant control over interruption timing does facilitate performance of the primary task (McFarlane, 1998), presumably because it allows the primary and interrupting tasks to be interleaved in ways that reduce disruption. Against this empirical background, it would be useful to have a theoretical framework to guide empirical research into a phenomenon as pervasive as being interrupted.

The goal-activation model suggests ways to improve experimental control over the effects of interruption by focusing on certain critical periods. Methodologically, the key implication of our fine-grained view of goal encoding is that the temporal structure of an interruption is richer than one might think. For example, a phone call comprises at least two events—the phone ringing, followed by the conversation that ensues when the callee picks up. Even a fire alarm consists of the ring itself, followed by the event of leaving the building. In such cases, the first event (the phone ringing, or the alarm sounding) can be viewed as an *alert,* and the subsequent event can be viewed as the *interruption* proper. The time between onset of the alert and onset of the interruption is the *interruption lag.*

The goal-activation model predicts that interruption lag is critical to the ability to resume an interrupted goal. Based on the priming constraint, the model predicts that efficient resumption depends on associative links between environmental cues and the target goal. Such links have to be formed before interruption onset (assuming that the interruption truly draws the system's attention away from the interrupted task). The interruption lag is a natural window of opportunity to form such links. By analogy, if two people are conversing and the phone rings, the callee faces a strategic decision—whether to use the available time while the phone is ringing to end the meeting with a few final words, or to reschedule it to continue at a later time. Similarly, the cognitive system may prefer to complete a goal if possible during the interruption lag, or link it to mental or environmental cues that facilitate resumption when the interruption is complete.

The processing of cues during the interruption lag may be relatively automatic, in which case simple manipulations of duration of interruption lag or the availability of cues should affect resumption efficiency. On the other hand, such processing may be deliberate, if it occurs at all, suggesting that it could be trained. That is, our model suggests that performance in dynamic task environments would be facilitated if operators were taught to react to an alert by searching for a cue and associating it with the goal being suspended. Moreover, the model has implications for the task environment itself—good cues need to be available during the interruption lag, and also at resumption, to allow first for associative encoding and then for associative priming.

In sum, the goal activation model suggests that one could manipulate the disruptiveness of an interruption experimentally, and reduce the disruptiveness of interruptions in real-world settings. Independent variables in such manipulations would include the relative timing of alert and interruption, the training of the participant or operator, and the extent to which the environment provides good cues at the right time. Moreover, the model suggests that the interruption lag is a critical period in which these manipulations will have their greatest effect, providing a focus for experimental control that has been lacking in previous studies of the effects of interruption.

6. Conclusions

The goal-activation model is a theory of how people remember their goals, or states of the world they want to achieve. The model shows that specialized memory structures are not necessary to explain goal-directed behavior in the Tower of Hanoi, a prototypical means-ends task that depends heavily on the suspension and resumption of goals. Goal-directed behavior can be explained with the general memory mechanisms of activation and associative priming. Activation initially comes from strengthening (or encoding, or "paying attention" to) the target goal, and is necessary to keep a goal active during any kind of planning or other mental simulation. Later, if the goal has been suspended and the time comes to resume it, associative priming is necessary to retrieve it from memory because it will be less active than goals encoded in the interim. Associative activation comes from cues linked to the target goal, cues that may lie in the environment or that may lie in long-term mental representations like procedural knowledge. Because such cues must be associatively linked to the target goal to be of any use as primes, cue availability and cue selection are important factors at goal-encoding time as well as at goal-retrieval time. An associated prediction is that experimental manipulation of cue availability at both encoding and retrieval should affect the ability to retrieve goals from memory.

The model suggests that intention superiority is an effect of opportunities afforded by the experimental procedure to improve performance of the target script. The model also suggests that interruption lag, or the interval between an alert and the interruption proper, is a critical period in which the cognitive system can prepare to resume the interrupted task. In future research, it will be important to examine whether and how the cognitive system makes use of the interruption lag and whether such behavior can be trained to help human operators manage goals more effectively in complex and dynamic task environments.

Notes

1. From Anderson and Lebiere (1998, p. 40): "In ACT-R, all of the goals on the stack are retained perfectly in order. If the current goal is popped, the next one is always there ready to take over. We have to confess to not being particularly sanguine about this feature. This assumption has not been stressed much because most cognitive tasks tend to have rather shallow goal structures (perhaps an indication that deep goal structures are problematical). One of the cognitive tasks that has the deepest goal structure is the Tower of Hanoi task—at least under certain strategies. As a later subsection displays, ACT-R has considerable success in modeling the available data from this task. Perhaps future research in this domain will expose problems with ACT-R's assumption of perfect goal memory."
2. The Base Level Learning Equation is $m = \ln(\sum_1^n t_j^{-d}) + \beta$ where t_j is the lag from the present of sampling event j, d is a decay parameter that is typically set to 0.5, and β is an initial-activation parameter that we set to 0. With these settings the equation simplifies to $m = \ln(2n/\sqrt{T})$ under the assumption that the t_j are evenly spaced. We then omit a factor of ln(2) to scale activation to be zero when $n = T = 1$, for reasons we describe in Note 6. This factor is safe to omit because it simply shifts the activation of all items upward independent of n and T and has no effect on the relationship between goal activation and the interference level.
3. The instructions given to participants included the following: "It is up to you to figure out how to solve the problems but the following hint may be helpful: As you work with the problems you will quickly realize that it is important to get the larger disks to their goal pegs first, since the ability to move a disk becomes more restricted as it becomes larger. It is a good idea to try to create a situation in which you can move the larger disks, and concentrate on the smaller disks only after this has been done." (Anderson et al., 1993, p. 126).
4. An instructional cue, indicating which of two scripts was to be acted out and which was to be observed, was displayed for two seconds. At that point a recognition memory test was administered using words from each of the scripts mixed with semantically-related foils. Recognition latency on hits was taken to index activation of the script.
5. The s parameter has a lower bound of 0 and usually ranges from 0.3 to 0.85 (Anderson & Lebiere, 1998, p. 217). The need for the interference level itself to be bounded suggests an additional upper bound on s of 0.5, which is a new theoretical constraint within ACT-R. However, we should note that 0.5 (which derives from the power of T in Eq. (1)) is actually the default value of another ACT-R parameter (d in the Base Level Learning Equation; see Note 2). We treated d as a constant because it seldom varies in practice, though the reason it seldom varies may be that s absorbs its variance.

Either way, the general version of the constraint implied by a bounded interference level is $s < d$.

6. Zero activation represents the default value of the ACT-R *retrieval threshold, τ,* which is a high-pass activation filter—if an item's activation falls below this threshold, the item is invisible to the system. If no element is above threshold when the system samples, that attempt fails and the system can either try again or move on to another activity. We take the retrieval threshold to coincide with trace activation at the time a trace is created in memory (before any strengthening). That is, a trace is created at threshold, and the system can "decide" at that point whether to invest in strengthening. The relationship between initial activation and retrieval threshold is not otherwise specified in ACT-R, so the assumption that they coincide is actually parsimonious in that it reduces two theoretical degrees of freedom to one, though it does require the scaling we described in Note 2. The assumption also has some reasonable implications. For example, it is consistent with the view that attention is the gateway to memory (Pashler, 1997)—for a memory element to persist, the system must allocate attention to it at encoding time (cf., Muter, 1980). In our model, this "attention" is simply the strengthening process. A computational description is given in Appendix B.
7. The activation noise we have been discussing is transient in that it applies to an element independently on every system cycle. ACT-R also contains a permanent activation noise that applies once to an element when the element is created and persists for the element's lifetime. Permanent activation noise allows representation of encoding-time variability, for example due to encoding-time manipulations that leave one item permanently stronger or weaker than another. The strengthening process in our simulation unpacks the encoding operation to which permanent activation noise applies. In the strengthening process, (transient) activation noise affects the test retrieval, which in turn affects strengthening time, which in turn affects the activation of the encoded element throughout its lifetime.

Acknowledgment

This work was supported by Air Force Office of Scientific Research grant F49620–97-1–0353 and by the Office of Naval Research. Portions of this work were presented at the twenty-first annual meeting of the Cognitive Science Society, Vancouver, Canada (August, 1999). Thanks to John Anderson, Larry Daily, Wayne Gray, Robert Rist, Michael Schoelles, Christian Schunn, Susan Trickett, Kurt VanLehn, and two anonymous reviewers for their comments and suggestions.

Appendix A. A formal model of strengthening time and the interference level

In the body of the paper, we argued that for a new goal to direct behavior, it had to be strengthened to stand out above the clutter of old goals in memory. The argument was based on the notion of the interference level—the activation of the most active distractor in

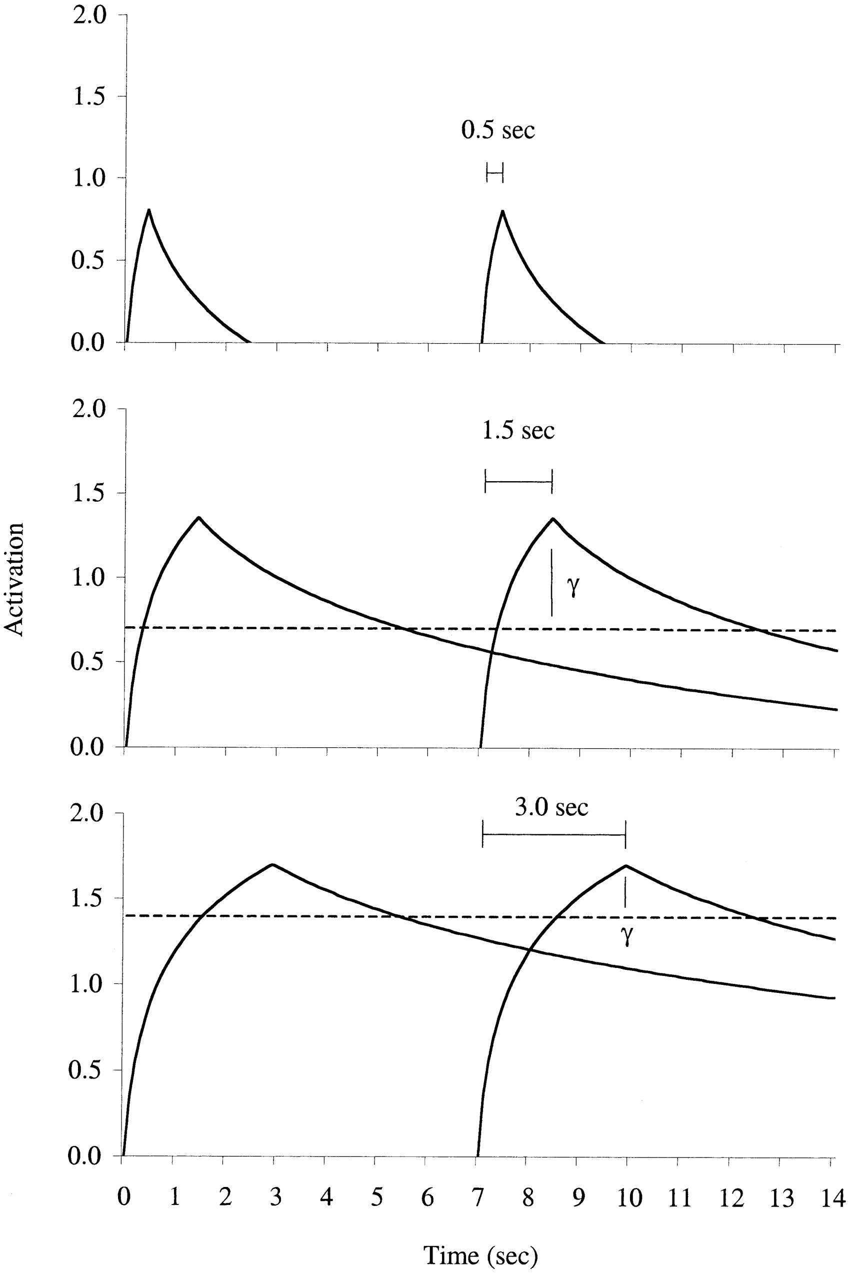

Fig. A1. Effects of strengthening on interference. The time course of activation is shown for two current goals. The interference level (dashed ink) is the activation of the most active old goal, and assumes that goals are encoded every 7 s and that each goal is strengthened by the same amount. Top panel: 0.5 s of strengthening fails to keep activation above zero (the retrieval threshold). Middle and bottom panels: More strengthening keeps activation above zero, but increases interference. The probability of sampling the new goal increases with γ, the amount by which the current goal is more active than the interference level.

memory. If the target is above this threshold then it is more likely to be sampled than any distractor, and if the target is below this level then it is less likely to be sampled than some distractor. The interference level depends on the number and strength of distractors—the more there are, or the stronger they are, the more they interfere with the target.

In this appendix, we develop an algebraic model of the interference level and apply it to the Tower of Hanoi data set discussed in the body of the paper. The purpose is to make concrete the notion that the duration of strengthening process—time to encode a goal—is bounded by functional constraints, both from above and from below. We will argue that some strengthening is necessary to raise a goal above the interference level and to keep it above the retrieval threshold (as distinct from the interference level), but that too much strengthening is counterproductive in that it raises the interference level for future goals. The purpose is also to test the model's quantitative predictions. From the Tower of Hanoi data, we estimate that one goal arises roughly every seven seconds. At this "goal tempo," the model suggests that one to two seconds of strengthening per goal is roughly optimal, which is consistent with empirical latencies on moves that require planning (i.e., generation of goals).

The first step is to formalize goal activation, from which we can determine the activation of each goal in memory (under certain assumptions) and thereby compute the interference level. In ACT, the activation m of a memory element fluctuates from cycle to cycle, with the magnitude of each fluctuation drawn from the logistic probability density:

$$p(\epsilon) = \frac{e^{-\epsilon/s}}{s(1 + e^{-\epsilon/s})^2} \tag{A1}$$

This density has mean zero (and standard deviation $\sigma = \pi s/\sqrt{3}$). ϵ is a random increment or decrement of activation added to the activation m of every memory element on every cycle, for the duration of that cycle. The probability of a particular value of ϵ is $p(\epsilon)$.

The logistic density allows a closed-form transformation that provides information about the extreme value of g logistic random variables (Johnson, Kotz & Balakrishnan, 1995). This transformation is useful here because there are g goals in memory, each with its own activation, and the most active of these (after adding ϵ) is what the system gets when it samples memory. Applied to ACT-R memory elements, the transformation is

$$m_\mathrm{a} = s\ \ln\left(\sum_{i=1}^{g} e^{m_i/s}\right) \tag{A2}$$

where m_a is the activation of the most active element and m_i is the activation of the ith individual element (Anderson & Lebiere, 1998). s is the variance parameter of the logistic density (Equation A1).

To make the interference level operational, we can apply this transformation to the old goals in memory–that is, m_i is the activation of the ith old goal, and m_a is the activation of the most active old goal. Thus, if the current goal is more active than m_a then it is more likely to be retrieved than any old goal. If the current goal is less active than m_a then it is less likely to be retrieved than some old goal. Note that more precise quantitative definitions of the interference level are possible. For example, one might say that a goal is above the interference level if its retrieval probability is 80% (to pick a number). However, defining it in terms of m_a is sufficient to let us analyze its sensitivity to changes in strengthening per goal.

Our next step is to estimate the m_i's in Equation A2, based on some assumptions. We assume that goals arise regularly in the course of performance, that there are an infinite number of them, and that each goal is processed (strengthened) in roughly the same way. Thus, the age of the ith old goal is $T = it$, where t is the *interencoding interval* (the interval between goal encodings). Substituting into Equation A1 and then into A2 produces

$$m_a = s \ln\left(\sum_{n=1}^{\infty}\left(\frac{n}{\sqrt{it}}\right)^{1/s}\right) \tag{A3}$$

The summation in Equation A3 is a p-series that converges if $s < 0.5$. We can estimate this series with an integral of the form:

$$\int_1^{\infty} \frac{1}{x^p} = \frac{1}{p-1},\ p > 1 \tag{A4}$$

Evaluating this integral with $x = i$ and $p = 1/(2s)$ produces a closed-form definition of the interference level:

$$m_a = \ln(n) - \frac{\ln(t)}{2} - s \ln\left(\frac{1}{2s} - 1\right) \tag{A5}$$

Equation A5 defines the interference level as a function of three parameters. The first is s (activation noise), which we set to 0.3. This is the value used in our simulation (see Appendix B), and satisfies the constraint that $s < 0.5$. The second parameter is t, the interencoding interval. We can estimate t for the Tower of Hanoi data by dividing solution time per problem by number of goals encoded per problem. Participants saw equal numbers of four- and five-disk problems, so we obtain solution time per "problem pair" by summing move latencies across the two problem sizes in Table 1. The result is 150 s (estimating 2.15 s for move 31 of the five-disk data). We then estimate the number of goal encodings per problem pair as the number of disks processed by the goal-recursion algorithm on moves 1, 9, and 13 of four-disk problems and moves 1, 17, 25, and 29 of five-disk problems. The total is 23, which we divide into 150 and round up, rather than down, to take some account of slack time between trials (which is time for old goals to decay and thus lowers the interference level). The result is an estimate of $t = 7$ s between goal encodings. The third parameter is n, the number of samples during strengthening. Strengthening causes activation to increase as governed by Eq. (1), repeated here:

$$m = \ln\left(\frac{n}{\sqrt{T}}\right) \tag{1}$$

When the system is fully engaged in strengthening, it can sample every 100 ms (under assumptions describing in Appendix B), which allows for a rapid increase in activation (m).

We can now examine the effect of strengthening on the interference level in concrete terms. Fig. A1, which elaborates on Fig. 1, shows the interference level for three levels of strengthening. The top panel shows that with $T = 0.5$ s ($n = 5$), the current goal falls below zero activation well before its term has expired. That is, the system loses track of the current goal several seconds before the next goal comes along.[6] Under these circumstances, behavior is not fully goal-directed. The bottom panel shows the opposite problem. With $T = 3.0$ s ($n = 30$), each goal has positive activation well into the future, but as a consequence the interference threshold is high from old goals that were similarly strengthened. (This was also the problem faced by the Byrne and Bovair, 1997, model of postcompletion error we discussed earlier, though there the cause of the problem was a lack of other memory elements to absorb the activation of the old goals.) The quantity γ allows a visual comparison. Larger values of γ imply more accurate memory for the current goal, because old goals are less likely to intrude. γ is clearly smaller with 3.0 s than with 1.5 s, showing that strengthening beyond a certain point actually degrades memory.

This analysis suggests that there is an optimal amount of strengthening for a given set of parameter values–about one to two seconds for the Tower of Hanoi data set. This estimate is intentionally vague because the model is based on a number of approximations and is undoubtedly incomplete. Nonetheless, the estimate is quantitative, and presents an opportunity to put the model to a test. Returning to the Tower of Hanoi data, latencies on planning moves (on which the goal-recursion algorithm runs) give an empirical estimate of encoding time per goal. This estimate is remarkably similar across the two problem sizes. For four-disk problems, the planning moves are 1, 7, and 9, and the sum of latencies on these moves is 21.7 s (see Table 1). Nine goals are generated (one for each disk smaller than or equal to the LOOP disk), so time per goal is 2.4 s. For five-disk problems, the planning moves are 1, 17, 25, and 29, and the sum of latencies is 33.7 s. Fourteen goals are generated, so time per goal is again 2.4 s. This span accommodates one to two seconds of strengthening time as well as time for the system to use that goal to formulate a subgoal. Thus, the model makes a ballpark prediction that is consistent with the data, and that accommodates distinct processes of goal storage (in memory) and goal formulation (Anderson & Douglass, 2001).

In sum, we made two main points in this appendix. First, strengthening time is restricted to a range by theoretical constraints–the retrieval threshold from below and the interference level from above. Second, we tested the predictions of these constraints on a particular data set, estimating the interencoding interval and from that deriving a rough estimate of strengthening time. An important next step will be to test the model on other data sets and in other domains. With its simplicity and relatively few parameters, one could imagine the model put to use as an engineering tool (cf. Gray, John, & Atwood, 1993; John & Kieras, 1996) for estimating the rate at which a human operator is able to update his or her goal in a dynamic task environment.

Appendix B. Details of the simulation

The body of the paper outlines the strengthening and priming constraints in general terms, but mapping these constraints to a running simulation required a number of additional assumptions and implementation decisions. We review some of these here, starting with the strengthening process for initializing the activation of a memory element. The ACT-R production system lacks a time-extended, information-processing account of how a memory element is encoded, and the strengthening process addresses that gap in what appears to be a general way. Next, we examine the cue-selection process, specific to the Tower of Hanoi, by which the model selects a cue from the task environment in order to retrieve a goal. The main cue-selection heuristic was introduced in the body of the paper, but here we give details and describe what happens when it doesn't apply. Finally, we discuss some other task-related procedural knowledge encoded in the simulation, and the simulation's parameters. Documented and executable code is available at http://www.msu.edu/∼ema/goals.

```
Strengthen-goal
    IF      disk D needs to move to peg P, and
            disk D is blocked, and
            D:P has not passed the retrieval test,
    THEN    encode another copy of  D:P in memory.

Conduct-tests
    IF      disk D needs to move to peg P, and
            disk D is blocked, and
            D:P has not passed the retrieval test, and
            D:P can be retrieved,
    THEN    conduct test retrievals of D:P.

Test-retrieval
    IF      test retrievals of D:P are being conducted, and
            D:P can be retrieved,
    THEN    note another successful retrieval.

Tests-pass
    IF      test retrievals of D:P are being conducted, and
            the desired number of test retrievals have succeeded,
    THEN    stop strengthening and move on to the next process.

Tests-fail
    IF      test retrievals of D:P are being conducted,
    THEN    stop test retrievals and continue strengthening.
```

Fig. B1. Pseudoproductions for the strengthening process and test retrieval in the Tower of Hanoi simulation. Control flow among these productions is shown in Fig. B2.

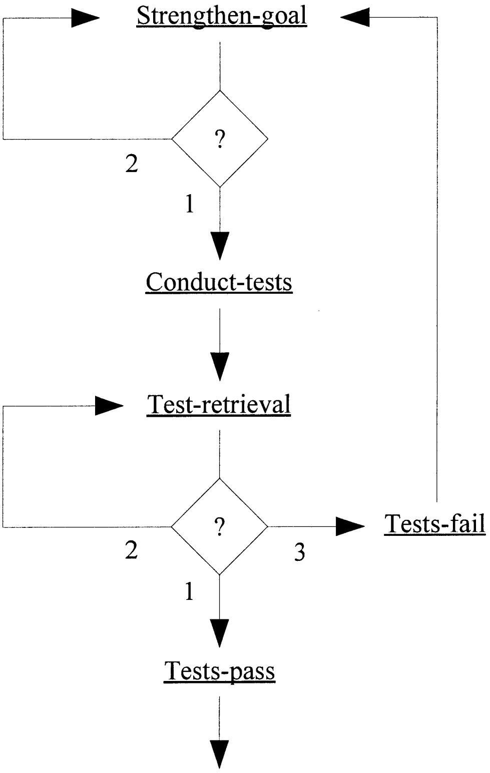

Fig. B2. Control flow among the goal-encoding productions of Fig. B1. An arrow indicates the next production selected to fire. A question mark ("?") indicates a conflict set containing multiple productions, and numbers indicate the order in which productions are selected from the conflict set. For example, after Strengthen-goal fires, Conduct-tests is selected to fire next, but if it cannot fire then Strengthen-goal is reselected.

B.1. The strengthening process

The strengthening constraint says that a new goal must be strengthened to direct behavior. Earlier we characterized the strengthening phase algebraically in terms of a rapid build-up of activation (Eq. (1)). Here we describe it computationally. There are two main aspects: An iterative process of sampling the item to increase its activation, and a criterion that specifies how much activation is enough.

ACT-R's production system integrates constraints like the Base Level Learning Equation with constraints on knowledge representation that let us estimate the duration of the strengthening process. ACT-R productions are fine-grained, in that they are quite limited in the amount of computation that an individual production can perform. Thus, a sequence of two production firings is required to encode a single declarative memory trace in memory. The first production gathers the information to be encoded and creates the new trace. The source of this information is the current goal, which has just been strengthened and so is now immediately and reliably available to any production that might need to use it. The second production in a coding cycle adds the newly encoded trace to declarative memory. When a new trace is added, it is merged with any other trace that is structurally identical, and the activation of the newly merged "master" trace is incremented in accord with the Base Level Learning equation. That is, the new trace disappears, and the master trace's *n* is incremented by one and its *T* incremented by the time elapsed since the previous merger (or retrieval, if there happened to be one). This merging mechanism is an analog of Logan's (1988) instance theory, in which encoding another trace for a previously-encountered item improves that item's availability in memory. Because there is a lag associated with a production firing, we can assign a duration to a coding cycle. The default firing time of one ACT production is 50 ms, so a two-production encoding cycle takes 100 ms. This estimate is used in Appendix A to derive a range estimate of one to two seconds of strengthening per goal for the Tower of Hanoi data set.

The second main aspect of the strengthening process is the criterion that says when strengthening should stop. Put another way, how does the system know how much activation is enough? In terms of Fig. 1, the question concerns the height of the activation peak, and (commensurately) the duration of the strengthening process. The analysis in Appendix A already provides considerable constraint on the duration of the strengthening process, by bounding strengthening functionally from above (by its effect on the interference level) and below (by proximity of the retrieval threshold). However, such constraints, implied by functional characteristics like Eq. (1), are distinct from the architectural processes that implement them.

Rather than develop separate a mechanism to monitor whether the functional constraints developed in Appendix A are being met, we simply built on existing mechanisms in the ACT-R production system. These mechanisms impose their own functional constraint: For a goal to be retrievable later, it must be active enough to be reliably above the retrieval threshold (as described in Appendix A) with priming from a retrieval cue factored in. This constraint translates into a simple test that the system can apply at encoding time to decide when to stop strengthening. The system simply needs to try to retrieve the goal, with the help of associative priming from the retrieval cue. If retrieval succeeds, this suggests that the goal is strong enough, and if the retrieval fails, this suggests that strengthening should continue. This *test retrieval,* as we will refer to it, is a kind of judgment of learning, though an automatic and architectural one rather than a deliberate self-report (e.g., Nelson & Narens, 1990). To have any predictive value, the test retrieval must factor in the effect of the retrieval cue, which in turn means that the retrieval cue must be available at encoding time. However, we have independent reasons to assume that the cue is available. The encoding process must have access to the retrieval cue to link it to the goal for use later when the goal is to be retrieved.

To prepare for the test retrieval, the system shifts its mental focus of attention to the retrieval cue. In operational terms, "shifting the focus of attention" to something in ACT-R means that all available associative activation is channeled through that thing to whatever targets it might be associatively linked to in memory. For the test retrieval, this means that all associative activation flows through the retrieval cue to the target goal. If this priming adds enough activation to make the goal retrievable *now,* this is an indication that the goal will be retrievable *later* (at resumption time). The goal will decay before then, of course, so a successful retrieval now overpredicts success later. To compensate for this overprediction, the test retrieval is "handicapped" by requiring that it pass multiple times in succession. (A given retrieval has some probability of failure, which is multiplied if a sequence of successful retrievals is required.) The extent by which the bar for the test retrieval is raised to account for decay is technically a parameter of the model that had to be estimated from the data. In practical terms, however, the model's performance is not particularly sensitive to how this parameter is implemented.

Pseudoproductions for the strengthening and test-retrieval processes are shown in Fig. B1, and the control flow among them is shown in Fig. B2. The core of the strengthening process is the strengthen-goal production, which initially fires in a loop, creating a new memory trace of the goal on each cycle. These traces are identical to one another, and ACT merges each trace with the first trace, or "master." As we described earlier, each merger increments the master's frequency count (*n* in Eq. (1)) by one, increasing the master's activation. Eventually the master is strong enough that conduct-tests fires to initiate the test-retrieval process. Two successive test retrievals, each with associative priming from the retrieval cue, must pass for the test-retrieval process as a whole to exit and allow the simulation to move on. If either test retrieval fails, control returns to the strengthening process.

The important points are that strengthening in the simulation is a process, rather than a parameter, and that the process is assembled from mechanisms existing already within the theory. The strengthening process makes a somewhat parochial contribution in that it eliminates the need for the permanent activation noise parameter in ACT-R.[7] The larger question, which no other activation-based memory theory we know of addresses with a computational account, is how to initialize a memory element's activation. Our strengthening process addresses this question in a way that applies to other kinds of goal encoding (Altmann & Gray, 2000) and may be useful as a model increment (Howes & Young, 1997) generally.

B.2. Cue selection in the Tower of Hanoi

Means-ends cues play a central role in the simulation, as they do in the model generally, because they supply the activation needed to retrieve an old goal. Such cues have to be attended at goal encoding, to create the associative link that will prime the goal later. They also have to be attended at goal retrieval, to actually prime the goal. We have no direct evidence that participants in the Tower of Hanoi study engaged in such processing, but we can ask whether it was possible. We identified the disks themselves as cues. At goal encoding time these have to be processed anyway, and at goal retrieval we described a simple perceptual heuristic that selects as a cue the disk uncovered by the last move. This heuristic usually works, but there are some exceptions, and here we describe these and how the model handles them.

The full set of rules for selecting a cue is shown in Fig. B3, and their flow of control in Fig. B4. The rule representing the heuristic we discussed earlier is Select-next-1. This is simply the inverse of the goal-recursion algorithm, applied after a move is made. Suppose that the cue selected by Select-next-1 is disk D. Previously, when the system was planning how to free D, it ran the goal-recursion algorithm "upward" by focusing on E (the disk on top of D). Thus, when E finally moves, the system runs the goal-recursion algorithm "downward," which suggests focusing on D. In simpler terms, the system tries to resume the supergoal once the immediate subgoal is achieved.

Two special cases arise in which Select-next-1 is insufficient. First, a move can empty a peg, rather than uncovering a disk. In this case, Select-next-2 selects the next larger disk than the one that was moved, wherever the next larger disk might be. This selection is heuristic, in that sometimes the next larger disk is not the best one to focus on next. However, focusing on the next larger disk is a good starting point, when combined with a rule (described next) that updates the selection when necessary.

The second special case arises when the model retrieves a "stale" goal—one that has been achieved but which has not been "replaced" in memory by a new goal for that disk. A goal becomes stale if its disk moves two or more times between planning phases. The model encodes a plan (or sequence of goals) once for each LOOP disk, but encodes only one goal per disk, even though some disks may need to move more than once to unblock the LOOP disk. For example, while unblocking disk 4, disk 2 moves twice (on moves 2 and 6, Fig. 4). When the goal for disk 2 is first retrieved (on move 2), it indicates the correct destination (peg C). When retrieved again (on move 6), the goal still indicates the same destination, but the disk is already there.

The Retrieved-stale rule recognizes when a disk is already at its destination, and responds

```
Select-next-1
      IF        the just-made move uncovered disk X,
      THEN      use X as the retrieval cue.

Select-next-2
      IF        the just-moved disk was X, and
                the move's source peg is now empty, and
                Y is one larger than X,
      THEN      use Y as the retrieval cue.

Retrieved-stale
      IF        a goal was retrieved for cue X, and
                the goal says to move X to a peg, and
                X is already at that peg, and
                Y is one larger than X,
      THEN      use Y as the retrieval cue.

Don't-undo
      IF        the just-moved disk was 1, and
                X is the smaller of the other two top disks, and
                Y is the larger of the other two top disks,
      THEN      move X on top of Y.

One-follows-two
      IF        the just-moved disk was 2,
      THEN      move 1 on top of 2.
```

Fig. B3. Pseudoproductions for the cue-selection heuristics (Select-next and Retrieved-stale) and the move selection heuristics (Don't-undo and One-follows-two) used to decide what to do next after making a move.

by selecting a new cue disk. As task-specific as this rule may seem, it follows directly from task constraints. A stale goal is easy to recognize: "This disk was supposed to go there . . . but it seems to be there already!" This recognition is a natural trigger for selecting another cue, and there is no point selecting a smaller disk, so the next larger disk is the natural choice.

B.3. Other task-related procedural knowledge

We have focused so far on the role of goals in guiding performance—the system encodes goals as it plans, and retrieves them later when the task environment hints that they may be achievable. However, memory is unreliable, and sometimes there is better information available to guide move selection. Here we examine two move-selection rules incorporated in the simulation that represent assumptions about procedural knowledge used by participants in the Tower of Hanoi experiment. These rules materially affect the behavior of the model and thus are important to assess for plausibility.

The two move-selection rules are illustrated in pseudocode form in Fig. B3. Because these heuristics interact with one another and with the cue-selection rules discussed above, they are included in the control-flow diagram in Fig. B4. The first move-selection rule, Don't-undo, says simply to avoid moving disk 1 two moves in a row. This heuristic is always correct and reasonably obvious, because two successive moves of disk 1 can always be combined into one with the same effect. When Don't-undo is in force, disk 1 moves every odd-numbered move, with larger disks moving on even-numbered moves. Don't-undo appears to be learned so easily that the actual learning event cannot be traced in protocol data (VanLehn, 1991) and requires no explicit instruction (Karat, 1982).

Don't-undo restricts disk 1 such that it never moves on even-numbered moves. Task-constraints further restrict the search space on even-numbered moves. With disk 1 excluded by Don't-undo, there are either one or two other movable disks to choose from. If there are two, then the smaller must move atop the larger, and if there is one then it must be moved to the empty peg. Thus, even-numbered moves are fully determined and require no reference to memory for goals.

The second move-selection rule, One-follows-two, applies to moves of disk 1 that follow moves of disk 2. Specifically, One-follows-two says that when disk 2 moves, disk 1 should be moved on top of it. One-follows-two is always correct for problems requiring 2^n-1 moves (as in the current data set), and like Don't-undo is quickly learned (VanLehn, 1991). However, One-follows-two is specific to this particular puzzle, whereas Don't-undo is based on general considerations of cognitive and movement efficiency. So, we assume that One-follows-two applies stochastically half the time, whereas Don't-undo applies always.

B.4. Simulation parameters

Our simulation generally uses default values for ACT-R parameters when they were available and values from other models when not. Default values were used for goal activation ($W = 1.0$), latency factor ($F = 1.0$), and base-level learning ($d = 0.5$). Transient activation noise ($s = 0.3$) and the value for ACT-R's retrieval threshold ($\tau = 4.0$) were taken from a task-switching simulation that uses the same strengthening process to encode a new

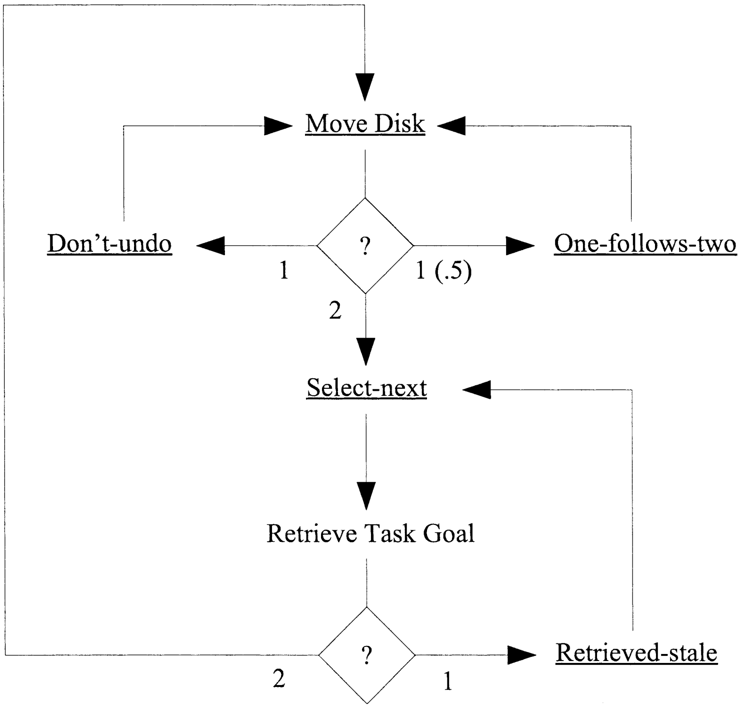

Fig. B4. Control flow among processes involved in making a move. The item at the head of an arrow is the next process to run, and underlined productions are those in Fig. B3. A question mark ("?") indicates a conflict set containing multiple processes. Numbers indicate the order in which processes are selected from the conflict set and attempted (and numbers in parentheses indicate selection probability, when less than one). For example, after Move Disk completes, either Don't-undo or One-follows-two is selected first, depending on which applies (they are mutually exclusive). One-follows-two is selected with probability 0.5. If neither applies, or if One-follows-two applies but is not selected, Select-next is selected.

goal for every switched-to task (Altmann & Gray, 2000). The amount of transient activation noise is consistent with other memory models implemented in ACT-R (Anderson & Lebiere, 1998) and satisfies the theoretical constraint from Appendix A that $s < 0.5$. The value for perceptual encoding time is 185 ms, which was taken from ACT-R models of menu scanning and the Sperling task (Anderson, Matessa & Lebiere, 1997). Baseline move time was set to 2.15 s, as in the model of Anderson and Lebiere (1998).

The value of 4.0 for the retrieval threshold is higher than the value of 0 we assumed in the algebraic model of Appendix A. However, this is not a conflict, because the threshold of 4.0 is for total activation, which is the sum of base-level activation and associative priming. In contrast, the value of 0 in Appendix A was for Base Level activation only. Moreover, a high retrieval threshold appears to prevent catastrophic error, in which one memory element comes to dominate all others. The activation dynamics in ACT produce a feed-forward, rich-get-richer situation, in which retrieval strengthens an item and thus makes it more likely to be retrieved on the next cycle. When a simulation involves thousands of retrievals, the system can quickly reach a perseverative state in which one memory element is so active that it effectively hides all other memory elements from the system. A sufficiently high retrieval threshold appears to prevent this, though a formal understanding of this mechanism has not yet been developed.

References

Altmann, E. M., & Gray, W. D. (2000). An integrated model of set shifting and maintenance. In N. Taatgen and J. Aasman (Eds.), *Proceedings of the third international conference on cognitive modeling* (pp. 17–24). Riethoek, The Netherlands: Universal Press.

Altmann, E. M., & Gray, W. D. (in press). Forgetting to remember: The functional relationship of decay and interference. *Psychological Science*.

Altmann, E. M., & John, B. E. (1999). Episodic indexing: A model of memory for attention events. *Cognitive Science, 23*(2), 117–146.

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

Anderson, J. R. (1990). *The adaptive character of thought.* Hillsdale, NJ: Erlbaum.

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

Anderson, J. R., Bothell, D., Lebiere, C., & Matessa, M. (1998). An integrated theory of list memory. *Journal of Memory & Language, 38*(4), 341–380.

Anderson, J. R., & Douglass, S. (2001). Tower of Hanoi: Evidence for the cost of goal retrieval. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 27,* 1331–1346.

Anderson, J. R., Kushmerick, N., & Lebiere, C. (1993). The Tower of Hanoi and goal structures. In J. R. Anderson (Ed.), *Rules of the mind* (pp. 121–142). Hillsdale, NJ: Erlbaum.

Anderson, J. R., & Lebiere, C. (Eds.). (1998). *The atomic components of thought.* Hillsdale, NJ: Erlbaum

Anderson, J. R., Matessa, M., & Lebiere, C. (1997). ACT-R: A theory of higher level cognition and its relation to visual attention. *Human-Computer Interaction, 12*(4), 439–462.

Anderson, J. R., & Milson, R. (1989). Human memory: An adaptive perspective. *Psychological Review, 96*(4), 703–719.

Anderson, J. R., & Reder, L. M. (1999). The fan effect: New results and new theories. *Journal of Experimental Psychology: General, 128*(2), 186–197.

Anzai, Y., & Simon, H. A. (1979). The theory of learning by doing. *Psychological Review, 86*(2), 124–140.

Arbuthnott, K. D. (1996). To repeat or not to repeat: Repetition facilitation and inhibition in sequential retrieval. *Journal of Experimental Psychology: General, 125*(3), 261–283.

Baddeley, A. D. (1986). *Working memory.* London: Oxford University Press.

Boyer, M., Destrebecqz, A., & Cleeremans, A. (1998). The serial reaction time task: Learning without knowing, or knowing without learning? *Proceedings of the twentieth annual meeting of the Cognitive Science Society* (pp. 167–172). Mahwah, NJ: Erlbaum.

Brandimonte, M., Einstein, G. O., & McDaniel, M. A. (Eds.). (1996). *Prospective memory: Theory and applications.* Mahwah, NJ: Erlbaum.

Byrne, M. D., & Anderson, J. R. (2001). Serial modules in parallel: The psychology refractory period and perfect time-sharing. *Psychological Review, 108,* 847–869.

Byrne, M. D., & Bovair, S. (1997). A working memory model of a common procedural error. *Cognitive Science, 21*(1), 31–61.

Carver, C. S., & Scheier, M. F. (1998). *On the self-regulation of behavior.* New York: Cambridge University Press.

Conway, A. R. A. (1999). The time-course of negative priming: Little evidence for episodic trace retrieval. *Memory & Cognition, 27*(4), 575–583.

Dagenbach, D., & Carr, T. H. (1994). *Inhibitory processes in attention, memory, and language.* New York: Academic Press.

Duncan, J., Emslie, H., & Williams, P. (1996). Intelligence and the frontal lobe: The organization of goal-directed behavior. *Cognitive Psychology, 30,* 257–303.

Egan, D. S., & Greeno, J. G. (1973). Theory of rule induction: Knowledge acquired in concept learning, serial pattern learning, and problem solving. In L. W. Gregg (Ed.), *Knowledge and cognition* (pp. 43–103). Hillsdale, NJ: Erlbaum.

Ernst, G. W., & Newell, A. (1969). *GPS: A case study in generality in problem solving.* New York: Academic Press.

Fum, D., & Del Missier, F. (2001). Adaptive selection of problem solving strategies, *Proceedings of the twenty-second annual meeting of the Cognitive Science Society* (pp. 313–318). Mahwah, NJ: Erlbaum.

Gillie, T., & Broadbent, D. (1989). What makes interruptions disruptive? A study of length, similarity, and complexity. Psychological Research, *50,* 243–250.

Goel, V., Pullara, D., & Grafman, J. (2001). A computational model of frontal lobe dysfunction: Working memory and the Tower of Hanoi task. *Cognitive Science, 25*(2), 287–313.

Goschke, T., & Kuhl, J. (1993). Representation of intentions: Persisting activation in memory. *Journal of Experimental Psychology: Learning, Memory, & Cognition, 19*(5), 1211–1226.

Gray, W. D., John, B. E., & Atwood, M. E. (1993). Project Ernestine: Validating a GOMS analysis for predicting and explaining real-world performance. *Human-Computer Interaction, 8*(3), 237–309.

Gunzelmann, G., & Anderson, J. R. (2001). An ACT-R model of the evolution of strategy use and problem difficulty. In E. M. Altmann, A. Cleeremans, C. D. Schunn, & W. D. Gray (Eds.), *Proceedings of the fourth international conference on cognitive modeling* (pp. 109–114). Hillsdale, NJ: Erlbaum.

Heath, C., Larrick, R. P., & Wu, G. (1999). Goals as reference points. *Cognitive Psychology, 38,* 79–109.

Howes, A., & Young, R. M. (1996). Learning consistent, interactive and meaningful device methods: a computational model. *Cognitive Science, 20*(3), 301–356.

Howes, A., & Young, R. M. (1997). The role of cognitive architecture in modelling the user: Soar's learning mechanism. *Human-Computer Interaction, 12*(4), 311–343.

Hutchins, E. L., Hollan, J. D., & Norman, D. A. (1985). Direct manipulation interfaces. *Human-Computer Interaction, 1*(4), 311–338.

Jeffries, R., Turner, A. A., Polson, P. G., & Atwood, M. E. (1981). The processes involved in designing software. In J. R. Anderson (Ed.), *Cognitive skills and their acquisition*. Hillsdale, NJ: Erlbaum.

John, B. E., & Kieras, D. E. (1996). The GOMS family of user interface analysis techniques: Comparison and contrast. *ACM Transactions on Computer-Human Interaction, 3*(4), 320–351.

Johnson, N. L., Kotz, S., & Balakrishnan, N. (1995). *Continuous univariate distributions* (Vol. 2). New York: Wiley.

Just, M. A., & Carpenter, P. A. (1992). A capacity theory of comprehension: Individual differences in working memory. *Psychological Review, 99*(1), 122–149.

Just, M. A., Carpenter, P. A., & Hemphill, D. D. (1996). Constraints on processing capacity: Architectural or implementational? In D. Steier & T. M. Mitchell (Eds.), *Mind matters: A tribute to Allen Newell* (pp. 141–178). Hillsdale, NJ: Erlbaum.

Kant, E., & Newell, A. (1984). Problem solving techniques for the design of algorithms. *Information Processing & Management, 20,* 97–18.

Karat, J. (1982). A model of problem solving with incomplete constraint knowledge. *Cognitive Psychology, 14*(4), 538–559.

Keppel, G., Postman, L., & Zavortink, B. (1968). Studies of learning to learn: VIII. The influence of massive amounts of training upon the learning and retention of paired-associate lists. *Journal of Verbal Learning and Verbal Behavior, 7,* 790–796.

Kimberg, D. Y., & Farah, M. J. (1993). A unified account of cognitive impairments following frontal lobe damage: The role of working memory in complex, organized behavior. *Journal of Experimental Psychology: General, 122*(4), 411–428.

Kotovsky, K., Hayes, J. R., & Simon, H. A. (1985). Why are some problems hard? Evidence from tower of hanoi. *Cognitive Psychology, 17,* 248–294.

Laird, J. E. (1984). *Universal subgoaling.* Doctoral dissertation, Computer Science, Carnegie Mellon University, Pittsburgh, PA.

Larkin, J. H. (1989). Display-based problem solving. In D. Klahr & K. Kotovsky (Eds.), *Complex information processing: The impact of Herbert A. Simon* (pp. 319–341). Hillsdale, NJ: Erlbaum.

Latorella, K. A. (1996). *Investigating interruptions: Implications for flightdeck performance.* Doctoral dissertation, State University of New York, Buffalo, NY.

Lebiere, C., & Lee, F. J. (2001). Intention superiority effect: A context-sensitivity account. In E. M. Altmann, A. Cleeremans, C. D. Schunn, & W. D. Gray (Eds.), *Proceedings of the fourth international conference on cognitive modeling* (pp. 139–144). Hillsdale, NJ: Erlbaum.

Lewin, K. (1926). Vorsatz, Wille und Bedurfnis [Will and needs]. *Psychologische Forschung* [*Psychological Research*], *7,* 330–385.

Logan, G. D. (1988). Toward an instance theory of automatization. *Psychological Review, 95*(4), 492–527.

Lovett, M. C., Reder, L. M., & Lebiere, C. (1999). Modeling working memory in a unified architecture: An ACT-R perspective. In A. Miyake & P. Shah (Eds.), *Models of working memory: Mechanisms of active maintenance and executive control* (pp. 135–182). New York: Cambridge University Press.

Marsh, R. L., Hicks, J. L., & Bink, M. L. (1998). Activation of completed, uncompleted, and partially completed intentions. *Journal of Experimental Psychology: Learning, Memory, & Cognition, 24*(2), 350–361.

Mayr, U., & Keele, S. W. (2000). Changing internal constraints on action: The role of backward inhibition. *Journal of Experimental Psychology: General, 129*(1), 4–26.

McFarlane, D. C. (1998). *Interruption of people in human-computer interaction.* Doctoral dissertation, George Washington University, Washington, DC.

Meyer, D. E., & Kieras, D. E. (1997). A computational theory of executive cognitive processes and multiple-task performance: Part 1. Basic mechanisms. *Psychological Review, 104*(1), 3–65.

Miller, G. A., Galanter, E., & Pribram, K. H. (1960). *Plans and the structure of behavior.* New York: Holt.

Monsell, S., & Driver, J. (Eds.). (2000). *Control of cognitive processes: Attention and performance XVIII.* Cambridge, MA: The MIT Press.

Muter, P. (1980). Very rapid forgetting. *Memory & Cognition, 8*(2), 174–179.

Nairne, J. S. (1992). The loss of positional certainty in long-term memory. *Psychological Science, 3*(3), 199–202.

Nelson, T. O., & Narens, L. (1990). Metamemory: A theoretical framework and new findings. In G. Bower (Ed.), *The psychology of learning and motivation* (Vol. 26, pp. 125–173). San Diego: Academic Press.

Newell, A. (1973). You can't play 20 questions with nature and win: Projective comments on the papers of this symposium. In W. G. Chase (Ed.), *Visual information processing* (pp. 283–308). New York: Academic Press.

Newell, A. (1990). *Unified theories of cognition.* Cambridge, MA: Harvard University Press.

Newell, A., & Simon, H. A. (1972). *Human problem solving.* Englewood Cliffs, NJ: Prentice-Hall, Inc.

Oaksford, M., & Chater, N. (Eds.). (1998). *Rational models of cognition.* New York: Oxford University Press.

Pashler, H. E. (1997). *The psychology of attention.* Cambridge, MA: The MIT Press.

Patalano, A. L., & Seifert, C. M. (1994). Memory for impasses during problem solving. *Memory & Cognition, 22*(2), 234–242.

Patalano, A. L., & Seifert, C. M. (1997). Opportunistic planning: Being reminded of pending goals. *Cognitive Psychology, 34,* 1–36.

Posner, M. I., & Cohen, Y. A. (1984). Components of visual orienting. In H. Buoma & D. G. Bouwhuis (Eds.), *Attention and performance X: Control of language processes* (pp. 531–556). Hillsdale, NJ: Erlbaum.

Powers, W. T. (1973). *Behavior: The control of perception.* Chicago: Aldine.

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.

Roediger, H. L., & Guynn, M. J. (1996). Retrieval processes. In E. L. Bjork & R. A. Bjork (Eds.), *Memory* (pp. 197–236). New York: Academic Press.

Ruiz, D., & Newell, A. (1989). Tower-noticing triggers strategy-change in the Tower of Hanoi: A Soar model. *Proceedings of the eleventh annual meeting of the Cognitive Science Society* (pp. 522–529). Hillsdale, NJ: Erlbaum.

Simon, H. A. (1975). The functional equivalence of problem solving skills. *Cognitive Psychology, 7*(2), 268–288.

Simon, H. A. (1992). What is an "explanation" of behavior? *Psychological Science, 3*(3), 150–161.

Steier, D. M., & Kant, E. (1985). The roles of execution and analysis in algorithm design. *IEEE Transactions on Software Engineering, SE-11*(11), 1375–1386.

Sternberg, S. (1998). Discovering mental processing stages: The method of additive factors. In D. Scarborough

& S. Sternberg (Eds.), *An invitation to cognitive science IV: Methods, models, and conceptual issues.* Cambridge, MA: The MIT Press.

Stretch, V., & Wixted, J. T. (1998). Decision rules for recognition memory confidence judgments. *Journal of Experimental Psychology: Learning, Memory, & Cognition, 24*(6), 1397–1410.

Suchman, L. A. (1987). *Plans and situated action: The problem of human-machine communication.* New York: Cambridge University Press.

VanLehn, K. (1991). Rule acquisition events in the discovery of problem-solving strategies. *Cognitive Science, 15,* 1–47.

VanLehn, K., & Ball, W. (1991). Goal reconstruction: How Teton blends situated action and planned action. In K. VanLehn (Ed.), *Architectures for intelligence* (pp. 147–189). Hillsdale, NJ: Erlbaum.

VanLehn, K., Ball, W., & Kowalski, B. (1989). Non-LIFO execution of cognitive procedures. *Cognitive Science, 13*(3), 415–465.

Waugh, N. C., & Norman, D. A. (1965). Primary memory. *Psychological Review, 72*(2), 89–104.

Young, R. M., & Lewis, R. L. (1999). The Soar cognitive architecture and human working memory. In A. Miyake & P. Shah (Eds.), *Models of working memory: Mechanisms of active maintenance and executive control* (pp. 224–256). New York: Cambridge University Press.

Zhang, J., & Norman, D. A. (1994). Representations in distributed cognitive tasks. *Cognitive Science, 18*(1), 87–122.

Ziegarnik, B. (1927). Ober das Behalten von erledigten und unerledigten Handlungen [On finished and unfinished tasks]. *Psychologische Forschung* [*Psychological Research*], *9,* 1–85.

Zijlstra, F. R. H., Roe, R. A., Leonora, A. B., & Krediet, I. (1999). Temporal factors in mental work: Effects of interrupted activities. *Journal of Occupational and Organizational Psychology, 72*(2), 164–185.