

# Doing Without Schema Hierarchies: A Recurrent Connectionist Approach to Routine Sequential Action and Its Pathologies

Matthew Botvinick

Department of Psychology, Carnegie Mellon University,  
Center for the Neural Basis of Cognition, and  
Western Psychiatric Institute and Clinic

David C. Plaut

Departments of Psychology and Computer Science,  
Carnegie Mellon University,  
and the Center for the Neural Basis of Cognition

In everyday tasks, selecting actions in the proper sequence requires a continuously updated representation of temporal context. Many existing models address this problem by positing a hierarchy of processing units, mirroring the roughly hierarchical structure of naturalistic tasks themselves. Although intuitive, such an approach has led to a number of difficulties, including a reliance on overly rigid sequencing mechanisms and a limited ability to address both learning and context sensitivity in behavior. We consider here an alternative framework in which the representation of temporal context depends on learned, recurrent connections within a network that maps from environmental inputs to actions. Applying this approach to the specific, and in many ways prototypical, everyday task of coffee-making, we examine its ability to account for several central characteristics of normal and impaired human performance. The model learns to deal flexibly with a complex set of sequencing constraints, encoding contextual information at multiple time-scales within a single, distributed internal representation. Mildly degrading this context representation leads to errors resembling the everyday “slips of action” that normal individuals commit under distraction. More severe degradation leads to a pattern of disorganization resembling that observed in *action disorganization syndrome*, a variety of apraxia. Analysis of the model’s function yields novel, testable predictions relevant to both normal and apraxic performance. Taken together, the results indicate that recurrent connectionist models offer a useful framework for understanding routine sequential action.

## Introduction

Much of everyday life is composed of routine activity. From the moment of getting up in the morning, daily living involves a collection of familiar, typically unproblematic action sequences such as dressing, eating breakfast, driving to work, etc. Because such activities can be executed without intense concentration, it is easy to overlook their psychological complexity. In fact, even the most routine everyday activities may call for a sophisticated coordination of perceptual and motor skills, semantic memory, working memory, and attentional control. Given the central role such activities play in naturalistic human behavior, it seems important to understand their psychological underpinnings.

Much existing work on sequential activity focuses on non-routine tasks, such as solving puzzles or playing chess

(Newell & Simon, 1972; Van Lehn, 1989), or has been concerned mainly with the question of whether sequence knowledge can be acquired implicitly (Stadler & Frensch, 1998). A small literature on ‘everyday,’ routine sequential activity has only gradually accumulated (e.g., Grafman, 1995; Humphreys, Forde, & Francis, in press; Miller, Galanter, & Pribram, 1960; Schank & Abelson, 1977; Schwartz & Buxbaum, 1997). Encouragingly, there appears to be growing interest in the domain, as evidenced by the recent appearance of several new studies using both experimental and computational modeling techniques (Cooper & Shallice, 2000; Humphreys & Forde, 1999; Riddoch, Edwards, Humphreys, West, & Heafield, 1998). These important efforts notwithstanding, it seems fair to say that the study of routine sequential action remains in its infancy.

In the present paper we advance, in basic form, a theory of how routine object-directed action sequences are performed. The framework we put forth, articulated in a set of computer simulations, takes as its point of departure existing work using recurrent connectionist networks. In applying such models to routine sequential action, the present work converges on two central theoretical claims: (1) The skills reflected in routine sequential activity cannot be identified with discrete, isolable knowledge structures, but instead emerge out of the interaction of many simple processing elements, each of which contributes to multiple behaviors; and (2) the detailed mechanisms that underlie routine action develop through learning, and as a result are closely tied to the

---

Financial support for this research was provided by NIMH Grant MH16804 and a grant from the American Psychiatric Association and Eli Lilly and Company to the first author, and an NIH FIRST award to the second author (Grant MH55628). The computational simulations were run using LENS (Rohde, 1999), <http://www.cs.cmu.edu/~dr/Lens/>. We thank Marlene Behrmann, Laurel Buxbaum, Myrna Schwartz and the CMU PDP research group for helpful comments and discussion.

structure of particular task domains. These tenets distinguish the present theory from other currently prevalent accounts, allowing it to overcome some of their limitations.

### *The Organization of Routine Tasks*

An essential point concerning the sequential structure of routine tasks was made early on by Lashley (1951). His aim was to point out the insufficiency of then current associationist accounts of sequencing, which characterized serial behavior as a chain of simple links between each action and the next. Lashley noted that because individual actions can appear in a variety of contexts, any given action may be associated with more than one subsequent action. In such cases, information limited to the action just performed provides an ambiguous cue for action selection. Lashley's conclusion was that, in addition to representations of individual actions, the actor must also have access to a broader representation of temporal context, a "schema" that somehow encodes the overall structure of the intended action sequence.

Later work has extended Lashley's argument by emphasizing that sequential action typically has multiple, hierarchically organized levels of structure (e.g., Miller et al., 1960; Schank & Abelson, 1977; Grafman, 1995). As stated by Fuster (1989, p. 159),

Successive units with limited short-term goals make larger and longer units with longer-term objectives. These in turn, make up still larger and longer units, and so on. Thus we have a pyramidal hierarchy of structural units of increasing duration and complexity serving a corresponding hierarchy of purposes.

An illustration, drawn from recent work by Humphreys and Forde (1999), is shown in Figure 1. Here, simple actions occurring during a typical morning routine (e.g., lifting a teapot) are grouped together into subroutines (e.g., pouring tea into a cup), which are themselves part of larger routines (e.g., making tea), and so forth. Rather than being organized by a unitary schema, behavior here has been described as involving the coordination of many schemas, associated with different levels of temporal structure.

### *Hierarchical Models of Action*

Since the original work highlighting the hierarchical structure of sequential behavior, a variety of proposals have been made concerning the cognitive mechanisms supporting such behavior. The most influential of these proposals share as a central assumption the idea that the hierarchical structure of behavior is mirrored in the gross architectural structure of the processing system. According to this approach, the processing system is arranged in layers corresponding to discrete levels of task structure, with processing at lower levels guided by input from higher ones. Models of this kind have proven partially successful in capturing basic phenomena relating to human sequential action. However, they also suffer from a set of important limitations. It is worth considering

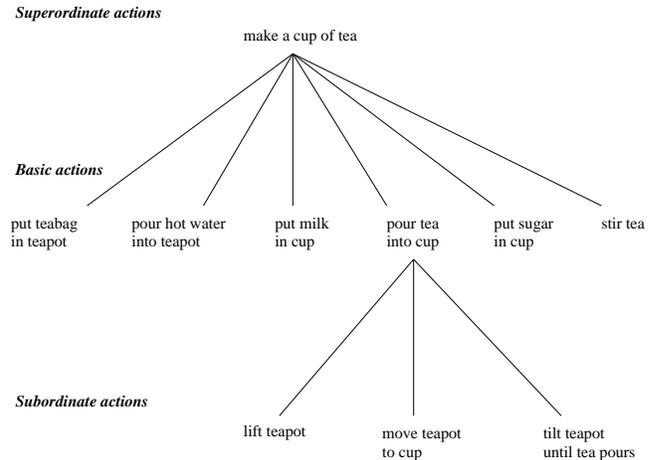


Figure 1. Hierarchical representation of a routine sequential task. Adapted from Humphreys and Forde (1999).

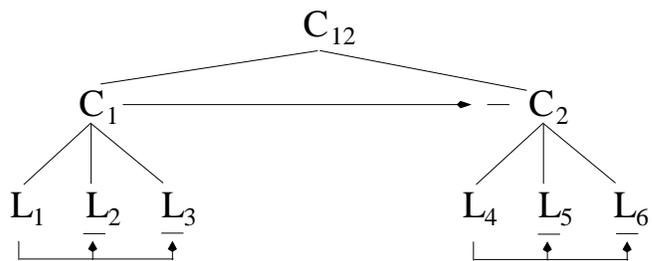


Figure 2. Diagram of the paradigm proposed by Estes (1972). Basic sequence elements are represented by the letter *L*, control elements by *C*.

hierarchical models in some detail, before turning to the alternative approach that will be the focus of the article.

One of the earliest attempts to model sequential action using a hierarchical processing architecture was by Estes (1972). As illustrated in Figure 2, Estes posited a hierarchy of "control elements," which activate units at the level below. Ordering of the lower units depends on lateral inhibitory connections, running from elements intended to fire earlier in the sequence to later elements. After some period of activity, elements are understood to enter a refractory period, allowing the next element in the sequence to fire. This basic scheme was later implemented in a computer simulation by Rumelhart and Norman (1982), with a focus on typing behavior. Here, nodes corresponding to individual words send activation to letter units. Letters coming earlier in the activated word inhibit later letters, until they enter a refractory period, as in Estes (1972).

Models proposed since this pioneering work have introduced a number of innovations. Norman and Shallice (1986) have discussed how schema activation might be influenced by environmental events; MacKay (1985, 1987) has suggested that action might be guided by a separate pool of "sequencing [or grammar] nodes," representing sequencing constraints separately from the units representing individual

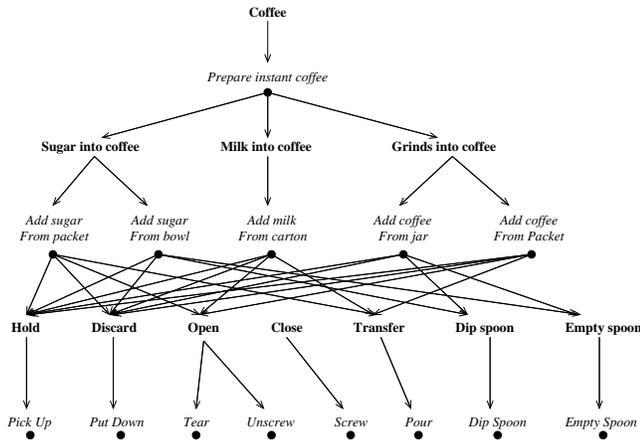


Figure 3. Processing architecture proposed by Cooper and Shallice (2000)

actions (see also Dell, Berger, & Svec, 1997); Grossberg (1986) and Houghton (1990) have introduced methods for giving schema nodes an evolving internal state, and have explored the consequences of allowing top-down connections to vary in weight, so that stronger activation is sent to units meant to fire earlier in a sequence than to units meant to fire later; and Cooper and Shallice (2000) have employed “goal nodes” that gate activation flow between levels. Importantly, though, despite these important developments, the majority of existing models continue to adopt the same basic approach established by Estes (1972). Contemporary theories of sequencing continue to assume that the hierarchical structure of sequential behavior is directly reflected in the structure of the processing system, as a hierarchy of nodes or schemas.

An illustration of the state-of-the-art is provided by Cooper and Shallice (2000). Their model, illustrated in Figure 3, addresses the everyday routine of making a cup of coffee. As in Estes (1972) and Rumelhart and Norman (1982), the processing system is structured as a hierarchy of nodes or units, with units at the lowest level representing simple actions (such as pouring or tearing), and nodes at higher levels representing progressively larger-scale aspects of the task. While the Cooper and Shallice (2000) model is in many ways more sophisticated than the one proposed by Estes (1972), the basic premise that behavioral hierarchy is mirrored in the structure of the processing system continues to organize the approach.

#### *Hierarchical Models and Action Pathology.*

One apparent strength of the hierarchical paradigm is that it offers an account not only of normal behavior, but also of action pathologies. Hierarchical models have been applied to error data from two domains, (1) everyday “slips of action” made by neurologically intact individuals (see, e.g., Baars, 1992; Cooper & Shallice, 2000; Norman, 1981; Reason, 1990; Roy, 1982; Rumelhart & Norman, 1982), and (2) the behavior of patients with ideational apraxia and action disorganization syndrome (ADS), neuropsychological disorders

involving impairments in performing sequential tasks using objects (described further below).

In most cases, only verbal descriptions have been given of how error data might be explained using a hierarchical processing framework. For example, Norman (1981) offered an account of action slips implicating faulty activation and triggering of schemas in his “activation-trigger-schema” framework. MacKay (1985, 1987) suggested that action slips might involve a weakening of the influence of high-level nodes on lower-level nodes in a hierarchical processing system. This work also proposed that ideational apraxia might be understood as involving a weakening of links among sequencing nodes. Schwartz, Reed, Montgomery, Palmer, and Mayer (1991) suggested that ADS might involve a weakening of top-down activation within a schema hierarchy.<sup>1</sup> Humphreys and Forde (1999) put forth a different hypothesis of ADS, suggesting that it might involve difficulties with establishing an activation gradient across units based on their intended firing sequence, as in the scheme proposed by Houghton (1990), or that it might stem from a breakdown in the mechanism causing units to become refractory after firing.

In addition to these verbal accounts, at least two studies have used fully-implemented computer models to address errors. Rumelhart and Norman (1982) showed that noise added to unit activations in their typing model led to errors resembling those of human typists. More recently, Cooper and Shallice (2000) showed that a variety of parameter manipulations in their hierarchical model can lead to errors resembling action slips. This study also addressed the behavior of ADS patients, modeling the disorder by weakening the influence of high-level schema nodes on lower-level nodes. Using this approach produced disorganized behavior, featuring many of the specific types of errors made by ADS patients.

#### *Difficulties with the Hierarchical Approach.*

Although the hierarchical approach has been successful in some respects, it suffers from several basic limitations. In the following sections, we focus on three particularly important problems: (1) failure to provide a satisfying account of how sequence knowledge is acquired, (2) failure to specify a sufficiently flexible sequencing mechanism, and (3) difficulty coping with behavioral domains that are not strictly hierarchical.

**Learning.** A first problem with the hierarchical approach is that it has proved difficult to specify a learning mechanism by which schema hierarchies might develop. A central impediment in this regard derives from hierarchical models’ reliance on localist representations, the use of single computational elements to represent supposedly discrete components of behavior. As illustrated above, such localist coding means that single units often code for composite structures. For example, a single computational element may be dedicated to representing the overall task of making coffee (see

<sup>1</sup> In subsequent writing from the Schwartz group, this hypothesis was substantially modified (see, e.g., Schwartz et al., 1995).

Figure 3). Since it is implausible to assume that a coffee-making schema is innate, hierarchical models are faced with the task of explaining how such a unit arises within the system. The discrete, localist nature of schema representation within such models has made it difficult to specify such an account. (We will consider one of the few existing attempts later in this article.)

**Sequencing.** A second problem with the hierarchical approach is the difficulty its proponents have encountered in specifying a sufficiently flexible sequencing mechanism. The earliest proposal, which involved lateral inhibitory connections, was unable to cope with situations in which the same items appear in more than one order across sequences (e.g., the letters *b*, *a* and *t* in *bat* and *tab*). This limitation led Rumelhart and Norman (1982) to insert a different set of weights into their typing model for the processing of each new word. MacKay (1985, 1987) took the alternate route of assuming separate representations for the same element appearing in different contexts (e.g., a unit for the *t* in *bat* and one for the *t* in *tab*). This remedy faces its own problems. First, the complexity of the resulting processing system makes it difficult to see how more than two levels of structure can be accommodated. Second, the use of multiple context-specific representations for each action raises concerns about the number of processing units that would be required to implement any realistic behavioral repertoire.

Given these difficulties, more recent models have abandoned the approach of using asymmetric lateral inhibition. Cooper and Shallice (2000) instead use “precondition gating” to guide the sequence of action selection. Here, activation flow to a particular action node is gated until appropriate environmental conditions are present. For example, the node representing “pour” in the sequence for adding cream to coffee is gated until the carton is open. As Cooper and Shallice admit, this approach faces difficulty in situations where environmental cues are ambiguous. For example, if one prefers coffee with sugar, one should wait until after adding sugar to begin drinking. However, since adding sugar does not change the appearance of the coffee, one cannot rely on environmental gating to inhibit triggering of the motor program for drinking. In order to deal with situations like this, Cooper and Shallice assume that activation flow to selected units is gated until the appropriate preceding actions have been completed (regardless of whether this is reflected in the environment). A problem with this “symbolic gating” is that, because no mechanism is specified for it, the model ends up assuming an important part of the functionality it is intended to explain.

Taking a different approach, Houghton (1990, see also Grossberg, 1986) has introduced compound units with time-varying state, which can be used to activate lower-level units in sequence. Despite the ingenuity of this approach, it faces a number of problems in dealing with sequencing behavior. First, while the approach has been used to address behavior with two levels of structure (e.g., spelling, involving word and letter levels), as with McKay’s proposal it is not obvious how deeper hierarchies might be coordinated. Secondly,

it is difficult to see how such a mechanism might deal with cross-temporal contingency, where choosing the correct action depends on retaining information about earlier actions. For illustration, consider the task of doing laundry using a washing machine. Here, in order to appropriately set the water temperature it is necessary to have some memory for which clothes one deposited into the machine earlier. It is not clear how such knowledge of earlier actions might be preserved and applied in the architecture described by Houghton (1990).

In Houghton’s model and others, an important mechanism in sequencing is *reflex (or rebound) inhibition*, by which units are automatically inhibited after firing (closely related to Estes’ refractory period). Far from an implementational expedient, reflex inhibition plays a direct explanatory role in work on negative priming (Houghton, 1994). When integrated into hierarchical models of action, this mechanism raises an implementational difficulty. The problem emerges at levels above the lowest level of the processing hierarchy. Here, it is important that units remain active until the segment of behavior they represent is completed. In the Cooper and Shallice (2000) coffee-making model, for example, the “add sugar” node must remain active throughout all of the steps involved in this subroutine. Immediate self-inhibition is thus inappropriate. In order to deal with this, Cooper and Shallice simply stipulate that units above the lowest level remain active until all relevant subgoals have been achieved. The mechanisms responsible for goal-monitoring and schema inhibition are, thus, left unspecified.

**Dealing with Quasi-Hierarchical Structure.** A final problem with traditional models involves their limited ability to cope with situations in which performance of a routine should vary with context. Although a given schema may be associated with multiple higher-level schemas (e.g., “add sugar” may be associated with schemas for making coffee and making a cake), there is typically no mechanism that allows the details of the lower level schema to change depending on the higher-level schema that recruits it.<sup>2</sup> In effect, the structure of hierarchical models enforces the assumption that tasks themselves are strictly hierarchical. There are, however, many everyday tasks that violate this assumption: A runner’s speed may depend on how long a race he is running; the water temperature one selects at the sink may depend on whether one is preparing to wash the baby or the dishes; the number of coins one deposits at a parking meter will depend on how long one intends to stay; and the care with which one brushes one’s hair in the morning may change depending on whether one is off to a job interview or to softball practice.

In order to see why such context-dependence presents a problem for hierarchical models, consider again the routine of making coffee, however this time as performed by a waiter serving three different regular customers. If each customer

<sup>2</sup> Some theories allow higher-level schemas to pass arguments to lower level schemas, allowing a degree of context-sensitive behavior. We point out some limitations of this “slot-filling” approach in the General Discussion.

prefers a different amount of sugar, the waiter will add a different number of spoonfuls depending on the particular customer he is serving at the moment. Modeling this coffee-making repertoire in a hierarchical model would present a dilemma. If different customers expect different amounts of sugar, should the sugar-adding routine be represented using one unit or several? Clearly, using one unit is inappropriate, since this does not allow the amount of sugar added to vary according to the individual being served. The alternative strategy of using several separate sugar-adding units ignores the fact that the different versions of sugar-adding are all instances of a single routine, and thus share a great deal of structure. The same dilemma arises at the level of the coffee-making task as a whole. Should the model represent coffee-making as a unitary schema, or as a set of independent schemas, each relating to a different customer? Because behavior in naturalistic tasks is often not strictly hierarchical, issues such as these pose an important problem for traditional models of action.

**Accounting for Error Data.** As noted earlier, hierarchical models have been employed to account for errors, both those of normal subjects and those of patients with apraxia. One recent effort along these lines, the modeling work of Cooper and Shallice (2000), is impressive in the range of data for which it accounts. Through various manipulations, Cooper and Shallice elicited errors in many of the major categories described in empirical studies, comparing their data explicitly with observations of apraxic patients reported by Schwartz et al. (1991, 1995).

However, despite its strengths, this recent work raises concerns about the ultimate ability of hierarchical models to account for error data in a comprehensive and parsimonious way. Cooper and Shallice found it necessary to use several different manipulations to produce different kinds of action slips. For example, while errors of omission were produced by weakening of top-down influence within the schema hierarchy, repetition or perseveration errors were attributed instead to insufficient lateral inhibition. In either case, it is not clear how the parameter manipulations made in the simulations relate to the conditions that are thought to predispose human subjects to action slips. In line with familiar assumptions, Reason (1984b, 1990) has argued that action slips tend to occur during periods of mental distraction. Cooper and Shallice do not explain how their parameter manipulations might be understood as a functional correlate of this.

Of greater concern is that the Cooper and Shallice (2000) model failed to capture at least two important features of the empirical data concerning errors. First, without the addition of special mechanisms, the model did not produce errors involving the repetition of an entire subtask after one or more intervening subtasks, a pattern often seen in slips of action. Second, the model did not reproduce an important relationship between error rate and error content in ADS, reported by Schwartz et al. (1998). This study found that, across patients, as overall error rate increased, omission errors formed a progressively higher proportion of all errors (see Figure 8

and further discussion below).

One further behavioral phenomenon that presents difficulty for hierarchical models is the fact that, when slips of action occur, it is often apparent how the preceding actions might have led the actor into the wrong program of action. Norman (1981) refers to such situations as instances of “capture,” and provides the following example: “[Soon after having played a game of cards], I was using a copy machine, and I was counting the pages. I found myself counting ‘1, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, King’ ” (p. 8). Another, well known example comes from William James (1890, p. 115), who describes how “very absent-minded persons in going to their bedroom to dress for dinner have been known to take off one garment after the other and finally to get in bed.” According to James, the bedtime sequence intrudes “because that was the habitual issue of the first few movements when performed at a later hour.” Because the representation of context in hierarchical models does not depend directly on which actions have been recently performed, it is unclear how such models might account for this aspect of errors.

### *An Alternative Approach*

An alternative framework for understanding sequential behavior is offered by existing work on *recurrent connectionist networks*. Beginning with the pioneering work of Jordan (1986) and Elman (1990), numerous studies have demonstrated the ability of such networks to produce sequential behavior resembling that of humans in a variety of respects. In the work to be reported here, we adapt the recurrent connectionist framework to the domain of routine, object- and goal-oriented sequential behavior, evaluating its ability to address a fundamental set of empirical phenomena.

The account we will put forth differs from the hierarchical approach in two basic ways. First, it does not attempt to build the structure of the task domain into the structure of the processing system itself. Instead, quasi-hierarchical, schema-like knowledge is acquired through exposure to the tasks the system is to perform. Second, information about sequential structure is not represented explicitly, but instead inheres in the emergent dynamical properties of the processing system as a whole. In the framework we will put forth, there is no isolable structure that can be identified with a schema. Borrowing the words of Rumelhart, Smolensky, McClelland, and Hinton (1986, p. 20),

Schemata are not “things.” There is no representational object which is a schema. Rather, schemata emerge at the moment they are needed from the interaction of large numbers of much simpler elements all working in concert with one another.

The knowledge that structures this interaction, in our account, is not represented locally as it is in hierarchical models. Instead, knowledge about a variety of action sequences is distributed in a superimposed fashion over a large set of connection weights among processing units. The result is a

system that displays behavior that can be hierarchically structured but also flexible and context-sensitive.

As we will show, the same properties that give such a processing system its power also make it susceptible to certain kinds of error. In learning to perform tasks with multiple levels of structure, the system must strike a balance between two demands: (1) the need to capture important distinctions among temporal or task contexts, and (2) the need to recognize patterns that recurring across tasks. For example, the system must be able to represent the fact that adding sugar to coffee is *both similar to and different from* adding sugar to tea. Within the account we will present, both pieces of information are captured in the same distributed representations. This superimposition of knowledge gives the system its ability to share information between tasks, to deal with quasi-hierarchical structure and, in principle, to generalize. However, it also places the system at risk for lapsing from one task into another, skipping ahead in a task, or incorrectly repeating its actions, in much the same way that both normal and apraxic humans do.

In the remainder of this section of the paper, we introduce the general computational framework, providing a basic description of how sequential networks operate and how they may be adapted to address human behavior in routine tasks involving the manipulation of objects. In the subsequent sections we enumerate a set of basic empirical phenomena relating to human sequential behavior, and then describe the results of a computer simulation study assessing the ability of a recurrent network model to address these empirical observations.

#### *The General Framework.*

Connectionist or parallel distributed processing models (Rumelhart & McClelland, 1986) comprise a set of simple processing units, each carrying a scalar activation value. The activation of each unit is based on excitation and inhibition received from units linked to it through weighted synapse-like connections. Often, the units in connectionist networks are segregated into three populations or “layers,” as illustrated in Figure 4. An input layer carries a pattern of activation representing some environmental stimulus. Activation propagates from this layer through an internal or hidden layer, which transforms the input information, sending a pattern of activation to an output layer whose units together represent the system’s response to the input.

A network is described as *recurrent* when loops or circuits can be traced through its set of connections. The network illustrated in Figure 4 contains such recurrent connections within its hidden layer, where each unit connects to every other as well as to itself. In the *simple recurrent network* (SRN) implementation (Elman, 1990; Jordan, 1986), these recurrent connections are associated with a transmission delay of one processing cycle.

To understand how recurrent connectivity supports sequencing, consider the operation of the network shown in Figure 4, over three successive time steps. On the first step, a stimulus pattern is imposed on the input layer. Activation from this layer propagates through the hidden layer, and re-

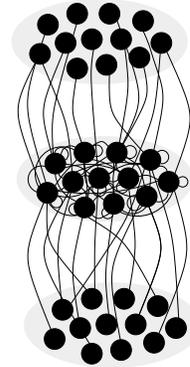


Figure 4. Schematic representation of the basic processing framework. For clarity, only a subset of the connections between units are shown.

sults in an output pattern in the network’s final layer. Importantly, in contributing to the transformation from input to output, the units in the hidden layer take on a pattern of activation that carries information about both the input and the output on the current step (Rumelhart, Durbin, Golden, & Chauvin, 1996).

On the next time step, a new pattern of activation appears in the input layer, and this again propagates to the hidden layer. On this step, however, the hidden layer actually receives *two* sources of input: In addition to activation from the input layer, the hidden layer also receives inputs by way of its own internal connections. The pattern of activation it receives via these connections is provided by the hidden layer representation on the last time step. Since this previous internal representation carries information about the input-output mapping on that time step, this source of input allows the network to combine current external inputs with information about its own previous activities.

The ability of the recurrent connectivity in the hidden layer to preserve information across time is by no means limited to a single time step. This is because, as the hidden layer combines information about external and internal sources of input, it takes on a new pattern of activation that reflects both. To continue the sequence we have been following, consider a third time step. Here, the hidden layer again receives inputs from both the input layer and the hidden layer pattern from the preceding time step. Because the latter pattern reflects events from both steps one and two, action selection on the current time step can be influenced by events occurring on either of these earlier steps. With further iterations of the processing system, action selection can be guided by events at increasing temporal distances.

The ability of such a system to map from inputs to appropriate outputs and to encode, preserve, and utilize information about temporal context depends on the pattern of connection weights among its units. Using a connectionist learning procedure such as back-propagation (Rumelhart, Hinton, & Williams, 1986), an effective set of weights can be learned through repeated exposure to correct sequential behavior. Through the gradual, adaptive adjustment of

its connection weights, the system learns to produce internal representations—patterns of activation across its hidden layer—that both facilitate the immediate selection of outputs and preserve information that will be needed later in the sequence.

*Previous Work on Recurrent Networks and Sequencing.*

The use of recurrence within connectionist networks to produce sequential behavior was first discussed by Jordan (1986). These studies employed feedback from output to hidden layers. Networks using feedback within the hidden layer were first studied by Elman (1990), who focused on linguistic phenomena. Since these initial studies, a number of other studies of sequencing in recurrent networks have appeared (e.g., Christiansen & Chater, 1999; Cleeremans, 1993; Dell, Chang, & Griffin, 1999; Dell, Juliano, & Govindjee, 1993; Elman, 1991, 1993; McClelland, St. John, & Taraban, 1989; Rohde & Plaut, 1999; Servan-Schreiber, Cleeremans, & McClelland, 1988). The basic properties of recurrent connectionist networks, as demonstrated in this existing work, suggest that they may offer an appealing alternative to hierarchical models in the domain of routine sequential action. Unlike hierarchical models, recurrent networks are intrinsically suited to sequential domains, containing a simple yet powerful mechanism for structuring behavior in time. Because recurrent networks can learn, there is no need to stipulate task-specific structure at the outset. Rather than having the necessary representations built in, recurrent networks learn to represent current and past events in a fashion suited to task demands (Elman, 1990; McClelland et al., 1989). Importantly, recurrent networks are capable of encoding temporal structure at multiple time-scales simultaneously (Cleeremans, 1993; Elman, 1991), pointing to a capacity to cope with hierarchically organized sequential structure. At the same time, as Elman (1991) has shown in models of sentence processing, recurrent networks do not suffer from the context insensitivity of hierarchical models, having the capacity to integrate information across multiple levels of temporal structure (see also Servan-Schreiber, Cleeremans, & McClelland, 1991).

Much of the existing research using recurrent networks to account for human sequential behavior addresses the domain of language processing. However, the processing principles at issue in this work are extremely general. Indeed, it has been proposed that these principles may have relevance to multiple domains of human behavior, including non-linguistic procedural skills (e.g., Gupta & Dell, 1999).

Despite the results of previous work with recurrent connectionist networks, widespread skepticism toward such models exists among researchers studying routine sequential action. For example, Houghton and Hartley (1995) have suggested that such networks necessarily suffer from the same limitations as “chaining” models. Brown, Preece, and Hulme (2000, p. 133) argue that recurrent networks lack “temporal competence. . . the intrinsic dynamics that would enable them to progress autonomously through a sequence.” Others have expressed specific doubt concerning the ability of recurrent networks to account for error data. For example, Cooper and

Shallice (2000, p. 329) write,

We know of no work in which such networks have been shown to be able to account for errors of the type observed in the action domain (specifically omission and other sequence errors). The principal difficulty in obtaining such errors within recurrent networks appears to arise from the lack of any separate representation of hierarchical relations (i.e., source/component schema relationships) and order information (i.e., the relative ordering of component schemas within a single source schema). It is thus difficult for order information to be disrupted without disruption to hierarchical relations.

Our work will attempt to demonstrate that such skepticism is misplaced.

*Modeling Naturalistic Action.*

The goal of applying the recurrent connectionist framework to routine sequential action raises several implementation issues. First, since everyday action typically involves action on objects, it is necessary to formulate a way to represent not only actions but also their targets and implements. Second, since actions often alter the perceived environment, it is necessary to allow this to occur in the model. Third, in approaching error data, it is necessary to motivate a technique for inducing dysfunction. In what follows, we detail our approach to these issues.

**Action on Objects.** Allport (1987, p. 395) has noted, “Systems that couple ‘perception’ to ‘action’ must deal moment by moment with two essential forms of selection: Which action? and Which object to act upon?” Because computational models of action have often dealt with tasks that do not involve direct physical action on objects (e.g., language tasks), they have typically focused only on the first of these two forms of selection. Thus, a central question facing models of routine naturalistic action is how objects are identified as targets for action.

One promising hypothesis is that targets for action are specified *indexically*. That is, actions are directed toward whatever object is currently at the system’s focus of orientation, where orientation can mean the point of visual fixation or, more generally, the focus of attention. This strategy, otherwise known as a “deictic” (Agre & Chapman, 1987; Ballard, Hayhoe, Pook, & Rao, 1997) or “do-it-where-I’m-looking” (Ballard, Hayhoe, Li, & Whitehead, 1992) strategy, has seen wide application in engineering and robotics (Whitehead & Ballard, 1990; McCallum, 1996). More importantly, it has been proposed as a model for how objects are selected as targets for action in human behavior (Agre & Chapman, 1987; Ballard et al., 1997, see also Kosslyn, 1994; Pylyshn, 1989; Ullman, 1984). The processing architecture shown in Figure 4 lends itself naturally to the use of indexical representation. One need only assume that the input layer, now interpreted as carrying a representation of

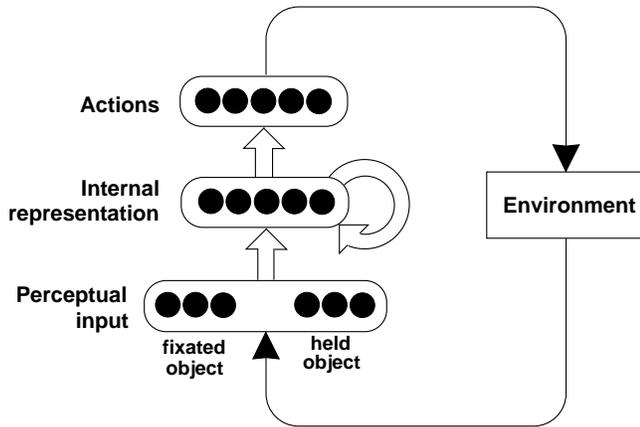


Figure 5. Architecture of the overall model. White arrows indicate that every unit in the sending layer is connected to every unit in the receiving layer. See text for details, including the number of units included in each layer.

the perceived environment, conveys information about which object is currently the focus of attention. Units selected in the model's output layer, now understood as representing actions, can be interpreted as directed toward that object. One potential implementation of this approach is diagrammed in Figure 5. Here, the input layer contains a segment labeled "fixated object," which specifies the visual features of the object currently at the focus of visual attention. The units in the output layer correspond to actions to be directed toward this object. Thus, if the input layer indicates that a coffee cup is currently fixated, and the action "pick up" is selected, the overall action is of picking up the fixated cup.

Some actions involve objects not only as targets but also as instruments or tools. To address this, the input layer in Figure 5 includes a second portion labeled "held object," which specifies the features of the object currently in hand. Just as the fixated object is interpreted as the target for action, the held object (if any) is interpreted as the implement to be used. For example, if the object fixated is a cup of coffee, the object held is a spoon, and the action selected is "stir," the overall action indicated is that of stirring the coffee using the spoon.

Because, within this framework, actions are directed at whatever object is currently the focus of attention, selecting a new target for action necessarily involves shifting that focus to a different object. To this end, computational models using indexical representations typically involve not only *manipulative* actions (actions that involve transformation of the environment), but also *perceptual* actions, which serve to reorient to the system toward a new object (see Whitehead & Ballard, 1990). This can be understood as either a physical reorientation, such as an ocular saccade, or a covert change of focus accomplished through attentional adjustments. Units representing such perceptual actions can be incorporated into the output layer of the architecture diagrammed in Figure 5, with each unit representing an action such as "fixate the spoon."

Given this framework, sequential action on objects takes the form of a rough alternation between perceptual actions, which orient the system toward a target object, and manipulative actions, where the object is acted upon. Evidence for such an alternation in human behavior is provided by Ballard et al. (1992). They monitored hand and eye movements in a block-moving task, finding that subjects alternated between perceptual and manipulative actions, saccading to a location in the workspace, acting at that location, saccading toward a new location, and so forth. An idealized description of this pattern of behavior is provided by Ballard and colleagues (p. 333):

- FIXATE (A block in the model area with a given color)
- REMEMBER (Its location within the context)
- FIXATE (A block in the source area with the same color)
- PICKUP (The block currently fixated)
- FIXATE (The appropriate location in the workspace)
- MOVE (The block to the fixated location)
- DROP (The block currently held at the current location)

A similar alternation between perceptual and manipulative actions has been observed in naturalistic tasks. For example, Hayhoe (2000) monitored eye and hand movements as subjects prepared a peanut butter and jelly sandwich. A portion of the pattern observed is diagrammed in Figure 6. As in the task used by Ballard et al. (1992), performance here involved an alternation between visual orientation toward a target object and manipulative actions directed toward that object (see also Land, Mennie, & Rusted, 1999).

Further support for the framework under discussion comes from neuropsychological work performed by Rid-doch, Humphreys, and Edwards (in press). These authors describe the behavior of a patient, ES, with anarchic hand syndrome, a condition that results from damage to the medial frontal lobe and corpus callosum and involves manual responses that appear impulsive and inappropriate and that are subjectively unwilling by the patient (DellaSala, Marchetti, & Spinnler, 1994). ES performed a task in which she was presented with two coffee cups, one green, the other red, and asked to pick up the green cup. She was instructed to grasp this cup by its handle, using her right hand if the cup appeared on the right of her body midline, and her left hand if on the left. ES rarely reached for the red cup, but did frequently reach with the wrong hand, typically when the handle of the target cup was oriented so as to invite this. Rid-doch et al. (in press) conclude, "ES is generally able to select the target object for action, even though she is impaired then at selecting the appropriate action for the target. This suggests that selection for action precedes the selection of action." This temporal and functional differentiation between perceptual and manipulative actions fits well with the indexical processing framework that we are adopting here.

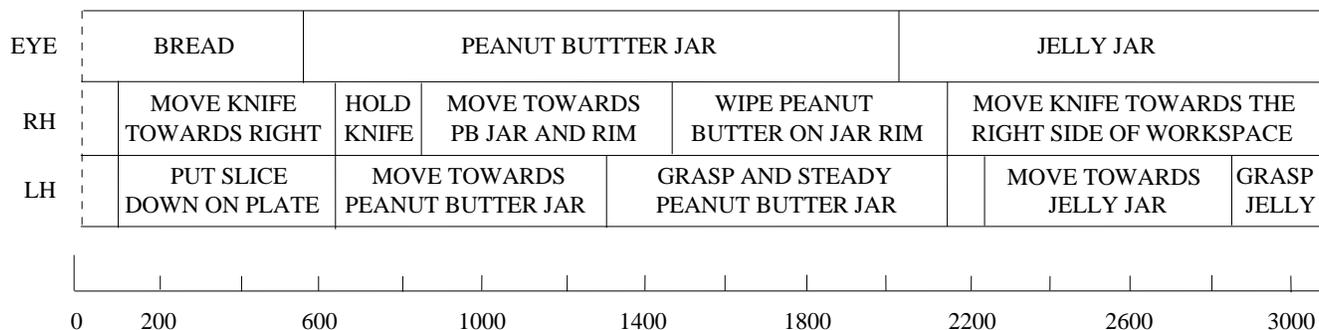


Figure 6. The pattern of visual fixations and manual actions observed by Hayhoe (2000) as subjects prepared a sandwich.

**Implementing the Perception-Action Loop.** An important aspect of naturalistic sequential action is that each movement, by altering the environment, can impact the perceptual input the system receives next. In order to capture this in a model, a functional representation of the environment must be interposed between the model's outputs and its subsequent inputs. The implementation diagramed in Figure 5 incorporates such a simulated workspace. This maintains a representation of the state of various objects in the environment, updates this in response to each action, and if appropriate yields a new input pattern to the layers representing the objects currently fixated and held. This portion of the implementation is also responsible for updating the fixated object in response to perceptual actions. It can therefore be thought of as conveying information not only about the environment, but also about the body's relation to the environment.

While some previous recurrent network models of sequencing have allowed model output to influence subsequent input, in general this has involved simply "copying" the model's output onto an input layer (e.g., Jordan, 1986; although see Plaut & Kello, 1999). Interposing a simulated environment, as in Figure 5, raises a number of computational issues not encountered in the simpler case. First, elements of the environment can serve as a form of external memory (Agre, 1988; Rumelhart et al., 1986). For example, adding cream to coffee changes the coffee's appearance, providing a visual cue that can potentially support later performance. Of course, not all actions leave such traces. Here, the processing system often must preserve information about hidden states of environment. We have already pointed out one example of this—adding sugar to coffee. Whitehead and Lin (1995, p. 281) provide another:

Consider a packing task which involves 4 steps: open a box, put a gift into it, close it, and seal it. An agent driven only by its current visual percepts cannot accomplish this task, because when facing a closed box the agent does not know if the gift is already in the box and therefore cannot decide whether to seal or open the box. In this case occlusion of the gift by the lid prevents immediate perception of a vital piece of information.

Another important feature of environmental feedback is that it may be probabilistic. To take an extreme example, consider a gambler operating a slot machine. Here, pulling the handle may yield any number of environmental outcomes, with subsets calling for different responses. Because there are many situations like this, where a particular action can yield multiple results, the action system must be ready to deal with environmental contingencies that are only partially predictable.

**Modeling Dysfunction.** Previous studies of action errors, in both normal subjects and individuals with apraxia, have regarded such errors as reflecting dysfunction in the basic mechanisms that give behavior its temporal structure. In hierarchical models, as we have discussed, this has involved weakening the top-down influence of high-level schemas, weakening lateral inhibition, and other manipulations. In the present framework, the most direct way to compromise the mechanisms that support sequencing is to disrupt the information carried by the recurrent connections within the hidden layer. Several different methods can be used to induce such a disruption. Previous studies using recurrent networks have added random perturbations to connection weights (Dell et al., 1993) or to net inputs (Cleeremans, 1993), or reduced the gain of the unit activation function (Cohen & Servan-Schreiber, 1992). In the modeling work to be reported here, we added random noise to the activation values being conducted over the processing system's recurrent connections.

Importantly, the disruption of internal representations in the present framework can be viewed as corresponding, in terms of its consequences, to basic etiological factors underlying both slips of action and ADS. Studies of slips have emphasized that such errors tend to occur during periods of distraction, where there is cross-talk from task-irrelevant cognitive activity (Reason, 1990). We assume that internal representations of temporal context are among the representations affected by such cross-talk. The addition of noise to these representations can thus be understood as a functional correlate of mental distraction. More severe levels of noise can be interpreted as representing the effects of direct neural damage in ADS. While the basic problem here is structural rather than functional (as in the case of slips), we assume that at the

computational level the two domains involve the same basic problem: a corruption of the system's internal representation of context. Interestingly, there is independent motivation for using a single technique for modeling both slips of action and errors in apraxia. Based on observations concerning the patterns of errors made by normals and patients with apraxia of varying severity, Schwartz et al. (1998) have suggested that apraxia may represent an exaggeration of the same processes that lead to errors in normals (see also Roy, 1982).

Using the approach just outlined, we conducted a series of computer simulations evaluating the capacity of a recurrent network, configured as in Figure 5, to account for a variety of basic phenomena relating to normal and disordered routine sequential action. In the next section, we detail the empirical phenomena to be addressed, relating to normal behavior, slips of action in normals, and the sequential behavior of apraxic patients. The simulation methods are then described, followed by a presentation of the modeling results in three sections, addressing behavior in each of these domains.

### Target Empirical Phenomena

Our simulations aimed to address a specific set of empirical observations relating to naturalistic sequential action. In this section we enumerate these, combining phenomena already discussed with several further benchmark observations.

#### *Normal behavior*

In contrast to the considerable effort that has been directed toward describing the structure of linguistic behavior, comparatively little empirical work has been done to characterize the structure of non-linguistic sequential action. Nonetheless, it is possible to specify a number of basic features that an account of sequential action should be able to explain.

1. As discussed earlier, routine sequences tend to assume a roughly hierarchical form. An overall task tends to be composed of distinguishable subtasks, which are themselves made up of distinguishable actions.

2. Elements at any level of this hierarchy may appear in multiple contexts. Individual actions may figure in more than one subtask, and a given subtask may feature in a number of different tasks. In many situations, it is impossible to produce correct sequences by "chaining" from one action to the next (Lashley, 1951).

3. Although the environment often provides cues to the correct action, the information it conveys is also frequently insufficient to guide action selection without some added context. The point is illustrated by the present-wrapping example from Whitehead and Lin (1995), cited earlier.

4. In some cases, it may be permissible to execute the elements of a sequence in variable order. For example, in preparing for bed, one may brush one's teeth and then wash one's face, or vice versa.

5. In some cases, actions or subroutines may be substituted for one another. For example, one may wash the dishes at the sink or by using the dishwasher.

6. The details of certain sequences may depend on the context in which they are performed, as in the example given above of the waiter making coffee with different amounts of sugar for different customers.

#### *Slips of Action*

Slips of action are the "absent-minded" mistakes normal individuals make from time to time while performing familiar tasks. While such errors have been of interest to psychologists since William James (1890, see also Jastrow, 1905), the most detailed information about the timing and form of such errors comes from work by James Reason (1979, 1984a, 1984b, 1990, 1992). Reason conducted diary studies in which subjects recorded the details of their own slips of action. In addition to Reason's own interpretation of this data, it has also been analyzed and extended by Norman (1981) and others (e.g., Baars, 1992; Roy, 1982). Such work points to a number of general principles that any sufficient model should address.

1. As noted earlier, slips tend to occur under conditions of distraction or preoccupation (Reason, 1990).

2. Slips tend to occur at "branch points" or "decision points," junctures where the immediately preceding actions and/or the environmental context bear associations with different subsequent actions. The following example is offered by Reason (1990, p. 70): "On passing through the back porch on my way to get my car I stopped to put on my Wellington boots and gardening jacket as if to work on the garden." Here, the behavioral and physical contexts relate to more than one activity, creating what Reason calls a branch point. The error involves a figurative (as well as literal) wrong turn at this point.

3. In the phenomenon Norman (1981) labeled "capture," lapses from one task into another tend to occur just after a series of actions that the two tasks share. The literature on action slips provides a wealth of examples of such errors, such as Norman's (1981) lapse from page-counting to card-counting or James' (1890) lapse from dressing for dinner into dressing for bed.

4. Slips tend to take the form of a familiar and intact sequence, ordinarily performed in a different but related context (Reason, 1979, 1984b).

5. Sequencing errors tend to fall into three basic categories: Perseverations, omissions, and intrusions (Reason, 1984a; object substitutions form a fourth major category of slip). Perseverations (or repetitions) occur when the sequence constituting the error derives from earlier within the same task. Intrusions occur when the sequence comes from a different, usually related, task. Omissions involve skipping over a subroutine to execute a sequence from later in the same task (e.g., "I wrote out a cheque and put the cheque book back into my bag without tearing the check out," Reason, 1984b, p. 535).

6. Slips involving lapses from one task into another (i.e., intrusions) tend to reflect the relative frequency of the two tasks. Specifically, lapses tend to involve a shift from a less frequently performed task into one more frequently per-

formed (Reason, 1979).

### Action Disorganization Syndrome

Impairment in performing everyday sequential routines, especially those involving the use of multiple objects, is frequently observed following brain injury. Most relevant to the issues under investigation here are two closely interrelated neuropsychological syndromes: ideational apraxia and frontal apraxia. While ideational apraxia is traditionally associated with left hemisphere lesions and frontal apraxia with prefrontal damage, in both syndromes patients display disorganization in their approach to sequential tasks (Duncan, 1986; Lehmkuhl & Poeck, 1981). In recent years, studies have challenged the specificity of ideational apraxia to left hemisphere damage (Buxbaum, Schwartz, & Montgomery, 1998), and have blurred the distinction between ideational and frontal apraxia. As a result, the anatomically neutral term action disorganization syndrome (ADS) has been adopted by some researchers (Schwartz, 1995).

Recent studies of ADS, most notably those performed by Schwartz and colleagues (Buxbaum et al., 1998; Schwartz, Mayer, Fitzpatrick, & Montgomery, 1993; Schwartz et al., 1998, 1995, see also Humphreys & Forde, 1999), have utilized explicit coding techniques to analyze patients' performance on naturalistic tasks both in the laboratory and in daily life, and have begun to yield a finer-grained picture of such patients' behavior. The principal findings can be summarized as follows:

1. Patients with ADS produce sequential behavior that is more fragmented than that of normals. Specifically, they show a tendency to abandon a subtask before the goal of that subtask has been accomplished. Schwartz et al. (1991) quantify this tendency by counting actions that occur outside the boundaries of completed subtask sequences, calling these "independent" actions. For illustration, consider the following sequence describing actions in the context of eating breakfast: The subject picks up a sugar packet, shakes it, opens it, and then puts it down in order to eat eggs. Because the actions with the packet would ordinarily be preparatory to pouring the sugar, and this did not occur before moving on to another activity, these actions would be counted as independents. Schwartz et al. (1991) evaluated the frequency of independent actions in the behavior of an ADS patient as he prepared instant coffee in the context of eating breakfast. In the earliest testing sessions, approximately one month after the onset of the disorder, roughly one half of the patient's actions were independents. Continued observations over the ensuing month revealed a gradual reduction in fragmentation as measured by the proportion of independent actions (Figure 7).

2. In addition to showing a general fragmentation of behavior, ADS patients commit frank errors, which fall into a characteristic set of categories. Most common are omission errors, for example neglecting to add sugar or cream when making a cup of coffee. Across studies, omissions have been consistently found to make up approximately 30–40% of ADS patients' errors (Buxbaum et al., 1998; Humphreys

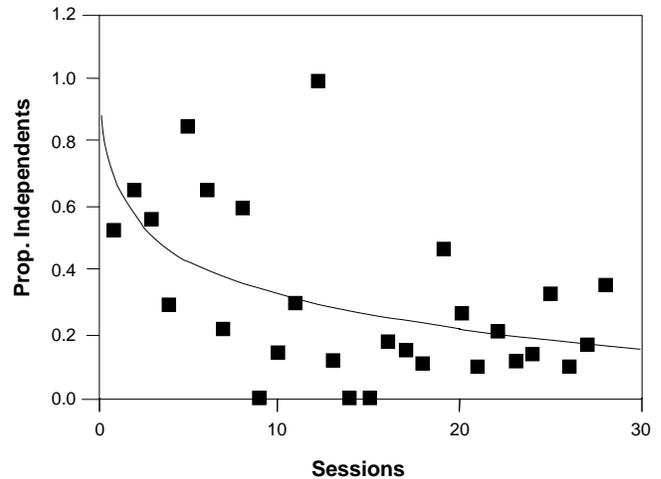


Figure 7. Proportion of independent actions produced by patient HH over the course of his recovery. (From Schwartz et al., 1991.)

- & Forde, 1999; Schwartz & Buxbaum, 1997; Schwartz et al., 1998). Next most frequent are sequencing errors, including repetitions of either of single steps or entire subtasks; anticipation errors, according to which an action is undertaken before a prerequisite action is completed, for example pouring a container of cream without having yet opened it; and (rarely) reversal errors, according to which two steps in a sequence are performed in the incorrect order. Sequencing errors, considered as a group, tend to make up approximately 20% of all errors. Other error types include action additions (actions that do not appear to belong to the assigned task), and substitution errors, according to which the correct action is performed using the wrong object, as when a patient stirs coffee using a fork, or when the wrong action is performed with the correct implement, as when a spoon is used to scoop up liquid from a cup of coffee, when the correct action would be to stir it. Schwartz et al. (1998) found that substitutions comprised 10% of errors, additions 12%. Less frequent error types observed in ADS include tool omissions (e.g., pouring sugar straight from a sugar bowl; 3% of errors in Schwartz et al., 1998), and quality errors (e.g., pouring far too much sugar into a cup of coffee; 8% of errors).

3. An important observation concerning omission errors in ADS is reported by Schwartz et al. (1998) and Buxbaum et al. (1998). As mentioned earlier, they found that, across patients, the proportion of omissions correlated with overall error rate (Figure 8). In mildly impaired subjects, omissions occurred with about the same frequency as sequence and substitution errors. In contrast, for subjects with the highest error rates, omissions formed a large majority of the errors committed.

## Simulations

Our computer simulations evaluated the ability of the recurrent connectionist framework to account for the empirical phenomena enumerated above. The simulations centered on

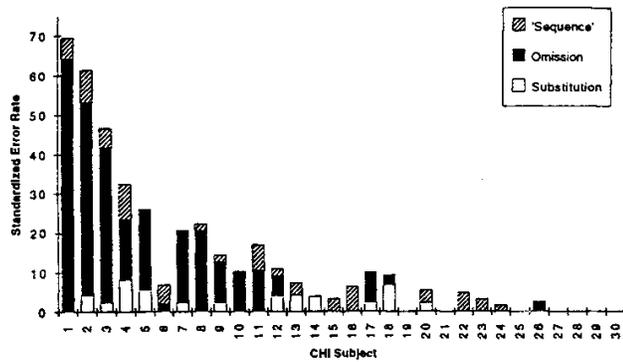


Figure 8. Data from Schwartz et al. (1998), showing the relationship between overall error rate and the distribution of error-types in a group of closed head injury patients. Each bar represents an individual participant.

a single model, trained on a set of concrete, everyday tasks, most centrally the task of making a cup of instant coffee. The behavior of this model was first compared with normal (error-free) human performance. Then, by impairing the mechanisms responsible for maintaining contextual information, the model was used to address slips of action and the behavior of patients with ADS. We also carried out two additional simulations, in which the model was trained on other training sets, in order to address the specific issues of context sensitivity and the effect of varying task frequency.

Following a description of the simulation methods, the presentation of results and analyses is organized into four subsections: task acquisition, normal performance, slips of action, and action pathology. For clarity, the details of the two additional simulations are described within the relevant results subsections.

## Methods

### Task and Representations.

The task of coffee-making provides a useful test case for modeling routine sequential action. First, it involves a flexible arrangement of sometimes interchangeable sub-tasks, displaying the kind of hierarchical structure that makes sequencing a computational challenge. Second, it is typical of the tasks discussed in the literature on slips of action in that it is familiar, performed without the need for highly focused attention, and—for many of us—a frequent occasion for absent-minded errors. Third, coffee-making is among the tasks that have been employed in the study of action disorganization syndrome (Schwartz et al., 1991, 1998; and see Lehmkuhl & Poeck, 1981). Finally, coffee-making has served as the focus for a recently proposed hierarchical model of sequential action (Cooper & Shallice, 2000). Addressing this task thus facilitates a comparison between approaches.

Our formalization of the coffee-making task is shown in Table 1. (As discussed below, the sequence shown corresponds to one of four versions of the task that were used in

Table 2  
*Object Features and Actions*

Fixated input	Held input	Action
cup	cup	pick-up
1-handle	1-handle	put-down
2-handles	2-handles	pour
lid	lid	peel-open
clear-liquid	clear-liquid	tear-open
light	light	pull-open
brown-liquid	brown-liquid	pull-off
carton	carton	scoop
open	open	sip
closed	closed	stir
packet	packet	dip
foil	foil	say-done
paper	paper	fixate-cup
torn	torn	fixate-teabag
untorn	untorn	fixate-coffee-pack
spoon	spoon	fixate-spoon
teabag	teabag	fixate-carton
sugar	sugar	fixate-sugar
instruct-coffee	nothing	fixate-sugarbowl
instruct-tea		

training.) The task consists of a series of discrete steps, each involving a set of perceptual inputs and an associated action. As introduced above, perceptual inputs are divided into those pertaining to the object currently viewed and those pertaining to the object currently grasped. In both cases, objects are represented as a collection of perceptible features. For example, a sugar packet that has been torn open is represented using the features packet, white-paper, and torn. The full feature set employed is listed in Table 2. A feature-based representational scheme was adopted in order to accommodate the fact that objects may share behaviorally relevant structure; in such cases we assume that there is overlap between objects' internal representations. At the same time, the use of high-level features circumvented difficult problems concerning perceptual encoding that were orthogonal to the issues of present interest.

As Table 2 indicates, viewed and held objects were associated with distinct feature sets. By convention, when the list of features describing the viewed object exactly matched that for the held object, this indicated that a single object was being held and viewed. When the descriptors for viewed and held objects differed, this indicated that one object was being held, and another viewed. (Table 2 also shows two "instruction" units, which will be discussed below.)

Also as shown in Table 2, actions were represented by single descriptors.<sup>3</sup> Actions fell into two broad categories: manipulative actions that alter the environment (e.g., pick-

<sup>3</sup> Actions were represented locally for the sake of convenience. In a more detailed implementation, to be described in a separate article, actions were represented—like objects—using feature vectors. Existing empirical work provides a preliminary basis for choosing

Table 1  
*The Coffee Task—One of Four Versions Used in Training*

Step	Fixated object	Held object	Action
1	cup, 1-handle, clear-liquid	nothing	fixate-coffee-pack
2	packet, foil, untornd	nothing	pick-up
3	packet, foil, untornd	packet, foil, untornd	pull-open
4	packet, foil, torn	packet, foil, torn	fixate-cup
5	cup, 1-handle, clear-liquid	packet, foil, torn	pour
6	cup, 1-handle, brown-liquid	packet-foil-torn	fixate-spoon
7	spoon	packet, foil, torn	put-down
8	spoon	nothing	pick-up
9	spoon	spoon	fixate-cup
10	cup, 1-handle, brown-liquid	spoon	stir
11	cup, 1-handle, brown-liquid	spoon	fixate-sugar
12	cup, 2-handles, lid	spoon	put-down
13	cup, 2-handles, lid	nothing	pull-off
14	cup, 2-handles, sugar	lid	fixate-spoon
15	spoon	lid	put-down
16	spoon	nothing	pick-up
17	spoon	spoon	fixate-sugarbowl
18	cup, 2-handles, sugar	spoon	scoop
19	cup, 2-handles, sugar	spoon-sugar	fixate-cup
20	cup, 1-handle, brown-liquid	spoon-sugar	pour
21	cup, 1-handle, brown-liquid	spoon	stir
22	cup, 1-handle, brown-liquid	spoon	fixate-carton
23	carton, closed	spoon	put-down
24	carton, closed	nothing	pick-up
25	carton, closed	carton, closed	peel-open
26	carton, open	carton, open	fixate-cup
27	cup, 1-handle, brown-liquid	carton-open	pour
28	cup, 1-handle, light-, brown-liquid	carton-open	fixate-spoon
29	spoon	carton-open	put-down
30	spoon	nothing	pick-up
31	spoon	spoon	fixate-cup
32	cup, 1-handle, light-, brown-liquid	spoon	stir
33	cup, 1-handle, light-, brown-liquid	spoon	put-down
34	cup, 1-handle, light-, brown-liquid	nothing	pick-up
35	cup, 1-handle, light-, brown-liquid	cup, 1-handle, light-, brown-liquid	sip
36	cup, 1-handle, light-, brown-liquid	cup, 1-handle, light-, brown-liquid	sip
37	cup, 1-handle, empty	cup, 1-handle, empty	say-done

up, pour, tear) and perceptual actions that orient the system toward a new object (e.g., fixate-cup, fixate-spoon). In keeping with the indexical representational scheme described above, the designations of manipulative actions did not specify which object was being acted on. This was determined entirely on the basis of which objects were currently viewed and held. Nonetheless, certain actions were intended to be appropriate to particular objects, and were associated with these within the training set. For example, tear was associated with the sugar packet, pull-open with the foil coffee-ground packet, and peel-open with the carton. Action des-

ignations pertained to observable sets of movements, rather than to associated task goals. Thus the actions tear, pull-open, and peel-open all serve the goal of opening a container, but were coded separately, reflecting their very dif-

which feature dimensions to use in such an enterprise (e.g., Klatzky, Pellegrino, McCloskey, & Lederman, 1993). One relevant set of parameters specifies the spatial coordinates toward which actions are directed. The present implementation avoids the important but difficult problem of specifying the spatial characteristics of actions. Results from preliminary modeling work suggests that the account presented here can be extended to address this problem.

ferent kinematic profiles.

In order to make the representational scheme as clear as possible, it is worth stepping through the first few steps of the coffee-making sequence shown in Table 1. In step one, the initial perceptual input is presented. The viewed object is the coffee cup, which contains only water. No object is yet grasped. Execution of the specified response for this step, *fixate-coffee-pack*, brings the coffee packet into view on step two. After execution of the pick-up action, the coffee packet remains in view but is also now grasped, providing the proper conditions for the pull-open action on step three. This, in turn, causes a switch from the *untorn* to the *torn* feature in the perceptual input for step four. Subsequent to this, *fixate-cup* brings the cup into view, a pour action results in the coffee grounds being added to the cup, turning the water brown, and so forth.

The sequence shown in Table 1 was one of four instances of the coffee-making sequence included in the training set. Each version of the task included four basic subtasks: (1) add coffee grounds to the hot water, (2) add cream, (3) add sugar, and (4) drink. However, the order of these subtasks varied. Specifically, in two sequences sugar was added before cream, and in the other two, cream before sugar. In addition, the training set also contained one subtask (sugar-adding) that appeared in two different forms. In two of the target sequences, sugar was added from a sugar bowl, in the other two, from a packet. Crossing these two dimensions of variability yielded the four exemplar sequences used.

GROUNDS → SUGAR (PACK) → CREAM → DRINK  
 GROUNDS → SUGAR (BOWL) → CREAM → DRINK  
 GROUNDS → CREAM → SUGAR (PACK) → DRINK  
 GROUNDS → CREAM → SUGAR (BOWL) → DRINK

The detailed structure of this coffee-making “repertoire” was designed to incorporate the basic characteristics of naturalistic routines, as enumerated earlier: (1) it involves hierarchical structure—individual acts aggregate into discrete subtasks that are themselves organized into one overall task; (2) it involves elements at both the action and subtask level that can appear in multiple contexts (e.g., the stir action, or the sugar-adding subtask); (3) it involves situations where environmental cues are not sufficient to guide action selection (e.g., the point just after adding cream, where the task requires a decision between adding sugar and beginning to drink); (4) elements may be executed in more than one sequential order (e.g., adding sugar and cream); and (5) other elements may be substituted for one another (e.g., either the sugar packet or the sugar bowl may be used for adding sugar).

At the same time, the coffee task was structured so as to facilitate a comparison between the model’s errors and everyday slips. As discussed earlier, slips of action tend to occur at branch points, junctures where the actions just performed bear associations with a number of different continuations. In the context of the coffee task, such branch points occur after the grounds-, sugar- and cream-adding sequences, each of which is associated with two different sets of subsequent events.

Several of the empirical phenomena we wished to address involve relations between two different tasks. For example, as discussed earlier, slips of action often take the form of a lapse from one task into another, especially when the two tasks in question share elements. In order to address phenomena such as this, it was necessary to train the network on a second task, in addition to coffee-making. The task chosen—tea-making—is detailed in Table 3. (A second version of the task, not shown, involved adding sugar from a sugar bowl rather than from a packet.) The structural relations between the coffee and tea tasks were designed to invite lapses from one task into the other. An important contrast between the two tasks is that, by convention, tea-making did not include a cream-adding subsequence. This feature made it straightforward to detect a lapse from tea-making into coffee-making; if cream was added to the tea, such a lapse had clearly occurred.

Note that the tea and coffee tasks begin with precisely the same perceptual input. This raises the issue of how the network was to “decide” which task to perform when this input is presented. As explained further below, in some simulations the choice of task was left up to the network. In other simulations, however, it was useful to have some means of directing the network to perform one task or the other. To this end, two additional input units were included: *instruct-coffee* and *instruct-tea*. These were intended to represent simply another form of perceptual input. While they can be thought of as representing auditory verbal commands, we included them with the visual input units, thinking of them as representing the visual cue cards used to instruct patients in some of the experiments of Schwartz et al. (1998). When used during training and testing, the instruction units were activated along with the initial perceptual input, and then inactivated. That is, the instruction units were only active during the first cycle of processing. Thus, while they provided an initial cue for action, they could bear none of the burden of representing task context on subsequent cycles of processing.

In addition to the tea and coffee tasks, the network was trained on an additional group of 267 single-step examples we refer to as the *background training set*. This contained one training example for each physically realizable configuration of the coffee-making environment. For each such input, the corresponding output included every action that might plausibly be performed in that environmental context, with unit activations normalized to sum to 1.0. For example, one example featured the input (*fixated: cup, 1-handle, empty; held: packet, brown-foil, torn*), with output activations of 0.125 for pour, put-down, and each of the fixate actions, and activations of zero for implausible or impossible actions such as stir, tear and sip. Each such training example is, in the limit, functionally equivalent to a much larger number of examples in which the same perceptual input is presented and each corresponding output consists of a single action sampled equiprobably from the set of possible actions (Elman, 1991). The purpose of adding the background set to the training corpus was to provide the network with information that might otherwise be derived from exposure to a wide variety of routines other than coffee- and tea-making, including information about basic physical constraints and affordances

Table 3  
*The Tea Task—One of Two Versions*

Step	Fixated object	Held object	Action
1	cup, 1-handle, clear-liquid	nothing	fixate-teabag
2	teabag	nothing	pick-up
3	teabag	teabag	fixate-cup
4	cup, 1-handle, clear-liquid	teabag	dip
5	cup, 1-handle, brown-liquid	teabag	fixate-sugar
6	packet, white-paper, unorn	teabag	put-down
7	packet, white-paper, unorn	nothing	pick-up
8	packet, white-paper, unorn	packet, white-paper, unorn	tear-open
9	packet, white-paper, torn	packet, white-paper, torn	fixate-cup
10	cup, 1-handle, brown-liquid	packet, white-paper, torn	pour
11	cup, 1-handle, brown-liquid	packet, white-paper, torn	fixate-spoon
12	spoon	packet, white-paper, torn	put-down
13	spoon	nothing	pick-up
14	spoon	spoon	fixate-cup
15	cup, 1-handle, brown-liquid	spoon	stir
16	cup, 1-handle, brown-liquid	spoon	put-down
17	cup, 1-handle, brown-liquid	nothing	pick-up
18	cup, 1-handle, brown-liquid	cup, 1-handle, brown-liquid	sip
19	cup, 1-handle, brown-liquid	cup, 1-handle, brown-liquid	sip
20	cup, 1-handle, empty	cup, 1-handle, empty	say-done

associated with objects.

#### *Model Architecture.*

The model architecture followed that shown in Figure 5. The input layer was divided into two major segments, one containing a unit for each feature describing viewed objects (20 units), the other with units corresponding to the features of held objects (19 units). This layer was fully connected to a hidden layer with 50 units, which was in turn fully connected to an output layer with a total of 19 units, one for each of the actions listed in Table 2. Each unit in the hidden layer was connected to all the other units in this layer, as well as to itself. Unlike the other connections in the system, these connections were associated with a conduction delay of one time step, thereby instantiating a simple recurrent network (Elman, 1990; Jordan, 1986).

As shown in Figure 5 and discussed above, the implementation included a mechanism to allow the model's output to influence its input. This included a simulation of the environment, implemented in Perl (Wall, Christiansen, & Orwant, 2000), which maintained information about the state of all objects (e.g., whether the carton was open or closed.) This segment of the implementation also kept track of which objects were currently viewed and held, updating the input to the fixated-object and held-object layers following pick-up, put-down or fixate actions.

For the primary model discussed in the simulations below, the training set included all four versions of the coffee task, both versions of the tea task, and the entire set of background examples. Each of the four coffee sequences

occurred twice during each epoch (pass through the training set): once with the instruct-coffee unit, and once without. Each tea sequence appeared four times, twice with the instruct-tea unit and twice without.

#### *Training Procedure.*

The network was trained on the target sequences using a version of recurrent back-propagation through time, adapted to the SRN architecture (see Williams & Zipser, 1995). Connection weights were initialized to small random values (sampled uniformly between  $\pm 1$ ). At the beginning of each training sequence, activations over the hidden units were initialized to random values (sampled uniformly between 0.01 and 0.99). On each time step, an input pattern, corresponding to a particular combination of viewed and held objects was applied to the input layer of the network (see Figs. 1 and 3). Activation was allowed to propagate through the network, producing a pattern of activation over the output layer. This output pattern was compared with the correct output for that step in the sequence (as defined by the target sequence). The difference between these two patterns, measured by their *cross-entropy* (see Hinton, 1989), was employed as a measure of performance error, providing the basis for gradual, adaptive weight changes. On the subsequent time step, the next input pattern from the training sequence was applied and the process was repeated. Weight updates were performed at the end of each individual sequence using learning rate of 0.001 and no momentum. Training was stopped after 20,000 passes through the training set, a point at which error levels had plateaued.

Note that, during training, the correct input sequence was presented to the network regardless of its generated outputs. This method of training (sometimes referred to as teacher-forcing) effectively encouraged the network to predict the next action in the sequence, analogous to a child watching an adult perform a task, continually anticipating what action will be performed next. During testing, however, the network's own generated output determined the next input, as described below.

Earlier work has shown that SRNs can have difficulty learning to preserve contextual information through sequences in which it is not immediately useful in selecting outputs (e.g., Servan-Schreiber et al., 1988). In order to support the present model's ability to preserve information over time, we introduced an additional term into the network's performance measure that pressured hidden unit activations to change as little as possible over successive processing steps.<sup>4</sup> Although this learning constraint was employed in all of the simulations reported here, we found in subsequent simulations that dropping the constraint yielded equivalent results.

#### *Testing Procedure.*

The model was tested by allowing it to produce sequences of actions without the external feedback provided during training. Prior to each test run, a random pattern of activation was applied to the hidden layer, as during training. On the first cycle of processing, the initial input pattern from the coffee and tea sequences was applied: (fixate: cup, 1-handle, clear-liquid; held: nothing). In some but not all of the simulations to be discussed below, the initial input also included activation over one of the two instruction units. Once activation had propagated to the output layer, the most active output unit was taken to specify the action selected. The representation of the environment was updated based on this action (even if incorrect), and used to generate a new input pattern.

During testing, feedback from the environment was probabilistic in two respects. First, the action fixate-sugar could result in either the sugar packet or the sugar bowl appearing as the viewed object on the next time step, with equal probability. Second, each sip action had a 50% chance of leaving the cup empty (whereas, during training, the cup always first appeared empty after two sips). The number of sips needed to finish the drinking sequence could therefore not be precisely predicted by the network.

The same set of connection weights was used in evaluating normal and impaired performance. In order to simulate the conditions involved in slips of action and ADS, the model's sequencing mechanism was disrupted by adding zero-mean normally distributed random noise to activation values in the hidden layer at the end of each cycle of processing. Low levels of noise (i.e., sampled from a distribution with small variance) were intended to correspond to the sort of mental distraction that attends everyday slips of action, or more technically, to cross-talk interference caused by cognitive activity occurring in parallel with the primary task. Higher levels of noise were intended to correspond to the result of physical

damage to the neural hardware involved in sequencing, as involved in ADS. With this in mind, it was hypothesized that low levels of noise would produce errors resembling human slips of action, and that higher levels of noise would produce behavior resembling ADS.

For the purposes of modeling error-free performance, a total of 300 test runs were completed with noise variance set to zero. In 100 of these, the instruct-coffee unit was included in the initial input pattern, in the next set of 100 the instruct-tea unit was included, and in the third set of 100 no instruction unit was activated. On each test trial, output was collected until the say-done action was produced. In order to model slips of action and ADS, 100 testing runs were conducted at each of the following levels of noise (variance): 0.02, 0.04, 0.08, 0.1, 0.2, 0.3, 0.4, and 0.5. Test runs began with the instruct-coffee unit activated, and were terminated when the say-done action was selected, or after 100 cycles.

#### *Evaluation of Performance.*

The behavior of the model in the absence of noise was evaluated by determining the degree to which the sequences produced matched those contained in the training set. Network performance at low levels of noise was compared with human slips of action. To this end, trials were separated into three categories based on whether they involved (1) no errors, (2) only subtask-level errors (errors occurring at branch-points and involving omission, repetition or intrusion of an entire subtask), or (3) at least one error that violated the structure of an individual subtask. Performance at higher levels of noise was compared with the behavior of patients with ADS. In order to compare model performance with the data on independents reported by Schwartz et al. (1991), the coding scheme used by these researchers was adapted to the coffee-making task as implemented in our training set, as detailed in the Appendix. Comparison of the specific error-types produced by the model with those of ADS patients was based on the classification specified by Schwartz et al. (1998), which includes object substitutions, gesture substitutions, action additions, tool omissions, quality errors, omission errors, and sequence errors comprising anticipation-omission errors, reversals, and perseverations.<sup>5</sup> Initial evaluation of the model's errors was directed at establishing whether the model produced examples of each of the above varieties of error. Quantitative analysis focused specifically on sequence and omis-

<sup>4</sup> In back-propagation, weight changes are made based on how they affect output error which, in turn, depends on how they affect unit activations. Thus, the procedure includes a computation of the derivative of error with respect to each unit's activation on each cycle. Our modification involved adding the term  $2\rho(a(t) - a(t-1))$  to the derivative for the network's hidden units, where  $a$  is the unit's activation and  $\rho$  is a scaling parameter (set to 0.05 in our simulations). This imposes a penalty on changing hidden unit activations that scales as the square of the difference between each hidden unit's activation on the present cycle and its activation on the previous one.

<sup>5</sup> Spatial misorientation and spatial misestimation errors are inapplicable given our non-spatial coding of action; these error types are also infrequent in the behavior of ADS patients (Schwartz et al., 1998).

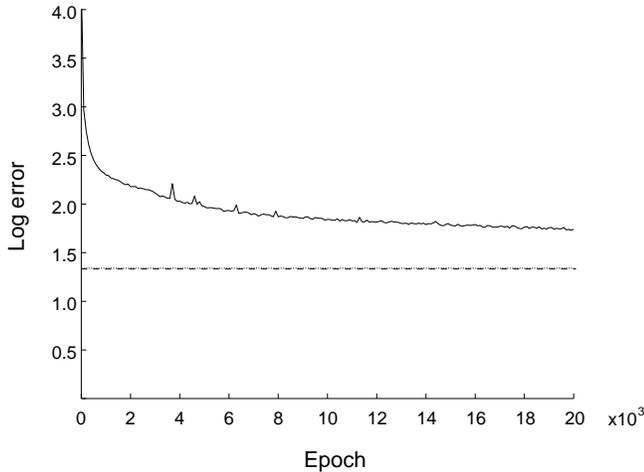


Figure 9. Log cross-entropy error for the entire example set over the course of training. Minimum achievable error, given variability inherent in the training set, was 22.18 (corresponding to a log of 1.34; dashed line).

sion errors. Enumeration of these errors was approached using a simplified version of the coding scheme employed by Schwartz et al. (1998), as described in the Appendix.

### Results: Task Acquisition

Figure 9 traces levels of output error over the course of training. By the point at which training was terminated (20,000 epochs), the network produced correct response predictions on every step of the training set. For all the sequences, output units fell within 0.1 of their target activation values. At the few points in the corpus where the subsequent action could not be uniquely predicted, unit activations roughly matched the probabilities of individual responses. For example, on the first step after completion of the add-grounds task, the action could be either fixate-carton and fixate-sugar. At test, the network produced activation values of near 0.5 for both of these actions.

Interestingly, the course of learning reflected a sensitivity to the hierarchical structure of the tasks in the training set. Specifically, error dropped more rapidly for steps falling within subtask boundaries than for steps falling at transitions between subtasks. The finding is illustrated in Figure 10, which shows the error incurred on each of the 37 steps of one of the coffee-making sequences, at four points in training. The pattern of interest is evident in the figure's solid trace, which shows the model's performance after 1000 passes through the training set. At this point, the error profile displays four prominent peaks, with relatively low error in between. As it turns out, the steps on which these peaks occur are the first steps of the four subtasks involved in the coffee task: grounds-adding, sugar-adding, cream-adding, and drinking. By 10,000 epochs of training, the model has learned to produce correct outputs at these transition points, as shown by the dashed line in the figure. In effect, the net-

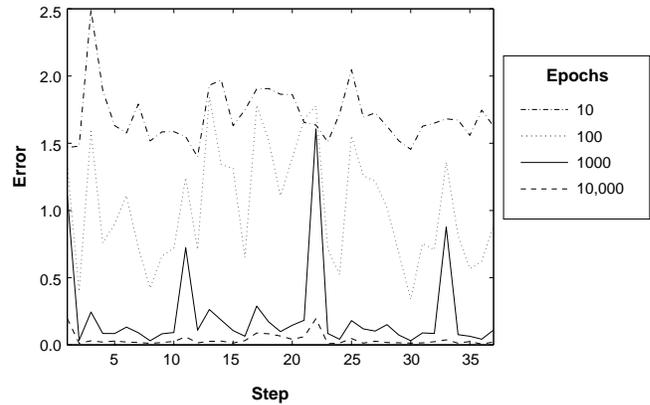


Figure 10. Average absolute error at each step of the sequence GROUNDS  $\rightarrow$  SUGAR (BOWL)  $\rightarrow$  CREAM  $\rightarrow$  DRINK at several points in training.

work learns the structure of subtasks earlier than it learns about the way that subtasks are organized into the overall task (see Elman, 1990; Hanson & Hanson, 1996, for related modeling results in other behavioral domains). It seems reasonable to assume that human learning proceeds along similar lines (see, e.g., Schwartz et al., 1991). However, we are aware of no experimental work directly demonstrating this (though see Greenfield, 1977). The progression from sub-task to task level structure seen in the model thus offers an empirically disconfirmable prediction about human learning.

### Results: Normal Performance

When the fully trained model was permitted to select actions without the feedback provided during training, it accurately reproduced the sequences from the training corpus. On each test run initiated with the instruct-coffee unit, the model produced one of the four coffee-making sequences. The number of occurrences of each variant of the task, in 100 test runs, is shown in Table 4. On each test run using the instruct-tea unit, the model produced one of the two tea sequences, as also shown in the table. The variability displayed in the model's behavior from trial to trial derives from the fact that a random pattern was imposed on the hidden layer at the beginning of each sequence. This random initial state was intended to serve as a proxy for internal states that would be present if the model were involved in a variety of other activities just prior to entering the coffee- or tea-making task. Importantly, production of the target sequences did not require inclusion of the instruction units. On test runs where no instruction unit was activated, the model produced one of the six training sequences, with the frequencies shown in Table 4.

Although the coffee and tea tasks may appear rather simple, it is important to note that they present the core set of computational challenges that hierarchical models have traditionally addressed. Indeed, our implementation of the coffee task closely resembles a version of that task used by

Table 4  
*Sequences Produced by the Model in the Absence of Noise*

With coffee instruction	
GROUND → SUGAR (PACK) → CREAM → DRINK	35
GROUND → SUGAR (BOWL) → CREAM → DRINK	37
GROUND → CREAM → SUGAR (PACK) → DRINK	14
GROUND → CREAM → SUGAR (BOWL) → DRINK	14
With tea instruction	
TEABAG → SUGAR (PACK) → DRINK	46
TEABAG → SUGAR (BOWL) → DRINK	54
With no instruction	
GROUND → SUGAR (PACK) → CREAM → DRINK	15
GROUND → SUGAR (BOWL) → CREAM → DRINK	18
GROUND → CREAM → SUGAR (PACK) → DRINK	12
GROUND → CREAM → SUGAR (BOWL) → DRINK	10
TEABAG → SUGAR (PACK) → DRINK	20
TEABAG → SUGAR (BOWL) → DRINK	25

Cooper and Shallice (2000). Like their task, the one used here requires the processing system to maintain and exploit an internal representation of task context. As an illustration, consider the sugar-adding subtask. Within the model’s training corpus, identical versions of this subtask appear as part of both the coffee task and tea task. Importantly, though, the sugar-adding subtask is followed by different actions in these two contexts. In the tea task, sugar-adding is always followed by the drinking routine. In the coffee task, however, sugar-adding can also be followed by cream-adding. Thus, even though the actions and perceptual inputs involved in sugar-adding are identical in the coffee- and tea-making contexts, it is necessary for the model to keep track of which task it is currently performing, in order to select the correct actions once sugar-adding has been completed.

In hierarchical models, this need to represent task context is typically met by including task-level schema nodes, e.g., the “prepare instant coffee” node in the Cooper and Shallice (2000) model. However, the performance of our model demonstrates that the same computational challenge can be met without assuming the existence of such discrete processing elements. A concrete illustration is provided by Figure 11, which shows the model’s behavior during the sugar-packet sequence, as performed in the context of both coffee- and tea-making. The left column shows the inputs to the model, determined “on the fly,” on the basis of the model’s action outputs. The two remaining columns show the outputs generated by the model during coffee-making (left) and tea-making (right). The last row in the figure shows the first step after the sugar-adding subtask has been completed, where the correct action is fixate-cream for the coffee task and put-down—the first step in the drinking sequence—for the tea task. The first thing to notice is that, for most of the steps shown, both the inputs to the model as well as its outputs are precisely the same for coffee- and tea-making. Once the model arrives at the last step shown in the figure, there is

nothing in its inputs or outputs during the preceding nine steps to indicate whether it has been involved in coffee- or tea-making. Nor is this indicated by the input on that final step. In order to choose the correct action on this step, the model must have access to an internal representation of task context. As the last row of the figure shows, the model is able to select the correct action in each context, indicating that it has discovered a way to represent the overall task setting.

In the foregoing instance, the network must maintain information about temporal context in order to avoid a lapse from one task into another. In other cases, context information is necessary to prevent actions from being omitted. For example, consider the version of the coffee task where cream is added just after the coffee grounds. Here, once the cream-adding sequence is completed, the network should enter into the sugar-adding subtask, by activating fixate-sugar. The external input to the model at this point is (fixated: cup, 1-handle, light-, brown-liquid; held: spoon). In other versions of the coffee task—where cream is added after sugar—this same input calls for put-down, the first step of the drinking subtask. Thus, taken on its own, the external input arising at the end of the cream sequence is ambiguous; it is associated with entry into both the sugar-adding and drinking subtasks. As a result, in the situation where cream has just been added and it is time to add sugar, the network must have some way to avoid prematurely launching into the drinking sequence, and thus inappropriately omitting the sugar sequence. In the hierarchical modeling framework, this would be accomplished through lateral inhibition or explicit preconditions. The model presented here stipulates no such mechanisms, yet its performance demonstrates that it was able to discover a means of avoiding the premature selection of actions. Specifically, it never made the error of entering the drinking subtask after cream-adding, when sugar was still to be added.

In still other cases, the network must access context information in order to prevent the incorrect repetition of actions. In the coffee task, for example, correct performance calls for adding sugar only once. This poses a challenge, since performing the sugar-adding sequence leaves no trace on the environment. Thus, the model must discover some means of keeping track of whether or not it has already completed sugar-adding, in order to avoid repeating the subtask. In most hierarchical models, such repetitions are prevented by a special mechanism that inhibits schemas once they have fired or once their associated goals have been accomplished. The present model does not build in any mechanism directed specifically at the prevention of repetitions. However, the trained model produced no repetition errors at all, indicating that it had discovered a means of keeping track of which subtasks had already been completed.

#### *Analyses.*

The model’s performance provides a basic demonstration of the ability of a recurrent connectionist network to cope with the central computational demands presented by a naturalistic sequential task, at least insofar as these are reflected in the structure of the coffee and tea tasks as we have implemented them. The model’s correct production of these

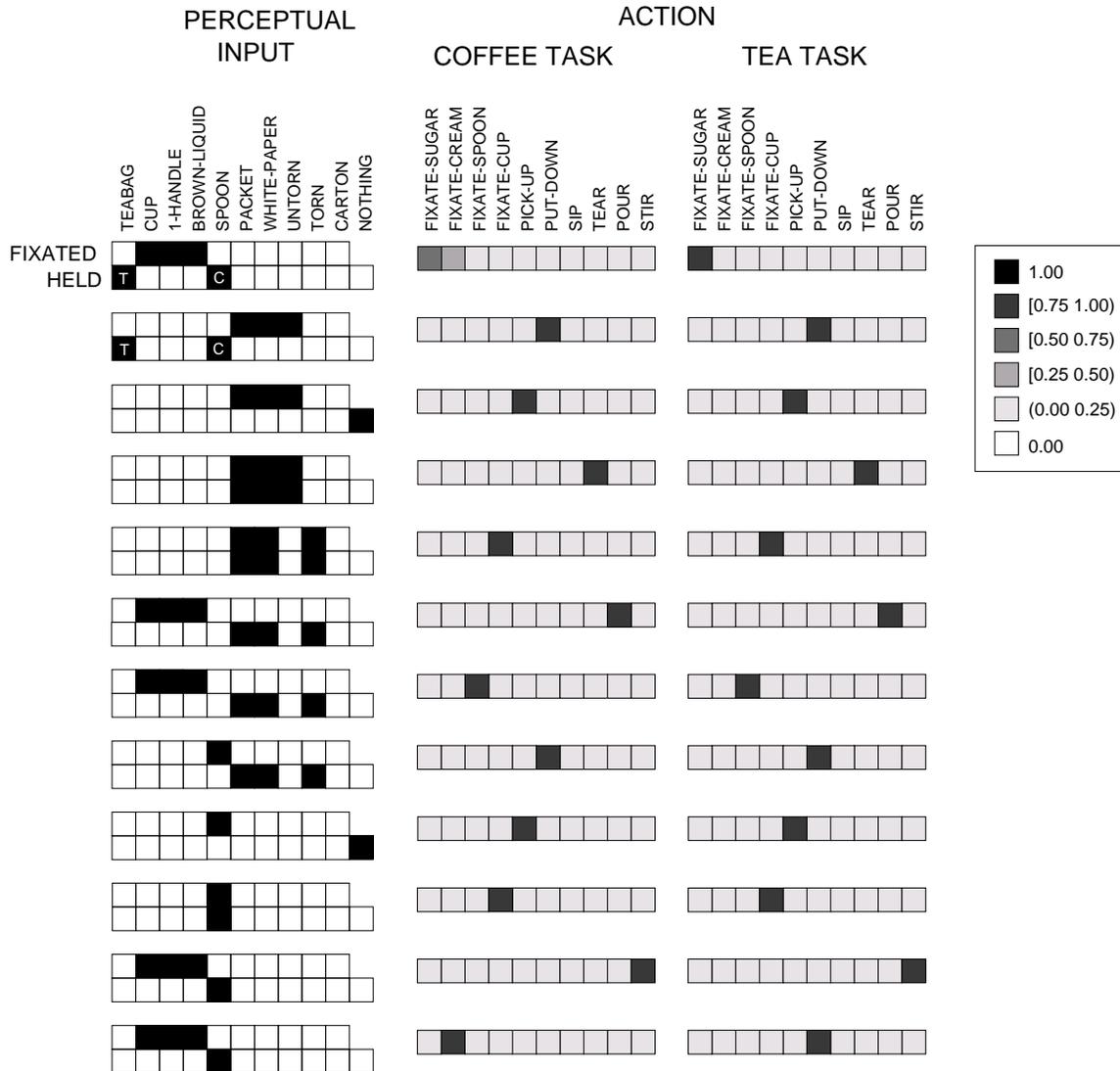


Figure 11. Unit activations during execution of the SUGAR (PACKET) subtask, as well as the first step of the next subtask, performed as part of coffee-making and tea-making (see Table 3, steps 5-16; for the coffee task, steps 12-23 are shown). The left column shows input patterns, generated in response to the network's output on the preceding step. Inputs are shown only once, since they are almost identical for the two tasks. Units active only in the coffee task are labelled with a C, units active only in the tea task with a T. The center column shows output unit activations in the coffee task, the right column output activations in the tea task. Units not selected during the sequence shown are omitted.

sequences indicates its ability to maintain information about task and subtask context and to use this information to guide response selection. Perhaps most importantly, it demonstrates that a hierarchically structured task can be managed in a processing system that is not itself structurally hierarchical. Rather than expressing the hierarchical organization of the task domain at the architectural level, the model captures this structure in the internal representations it uses to perform the task. Because these representations are the key to understanding the behavior of the model, it is worth considering them in some depth.

On each step of processing, the model's hidden layer assumes a new pattern of activation. Since this pattern reflects

both the current stimulus-response mapping and any contextual information being preserved, it can be thought of as a compact and context-specific representation of the current step in the task. As there are fifty units in the hidden layer, this internal representation can be represented as a point in a fifty-dimensional *state space*. As the task sequence unfolds and the model's internal representation evolves, a trajectory is traced through that space. As Elman (1991, 1993) has shown, it is useful to visualize such trajectories as a way of understanding how recurrent models represent tasks they have learned to perform. One useful way of accomplishing this is through multidimensional scaling (MDS). MDS yields a representation in two dimensions that preserves as much

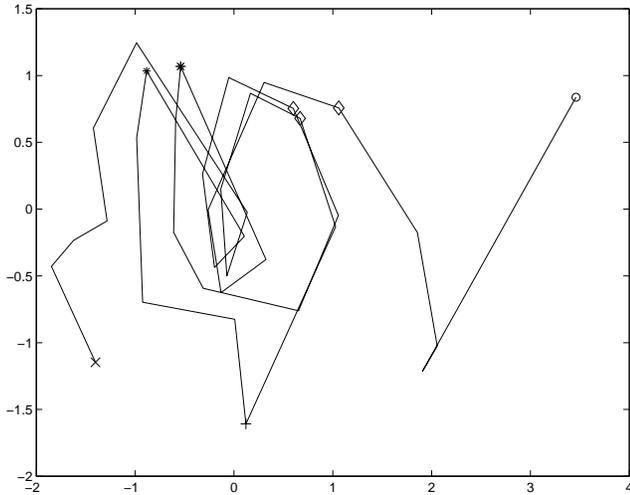


Figure 12. MDS plot of hidden representations over the course of the sequence GROUNDS → SUGAR (PACK) → CREAM → DRINK. The trajectory begins to the far right.  $\diamond$ : pour actions,  $+$ : peel-open carton,  $*$ : two stir actions (see text). The characters 'o' and 'x' indicate beginning and end of sequence, respectively (and this convention applies throughout the following MDS diagrams).

information as possible about the original distances among a set of points in a higher dimensional space (Kruskal & Wish, 1978). An illustration of MDS applied to the present model's internal representations is presented in Figure 12. This shows the trajectory followed during the entire coffee-making sequence, GROUNDS → SUGAR (PACK) → CREAM → DRINK. The individual points represent the hidden representations arising on each of the 37 steps of the sequence. The line segments connecting the points indicate their arrangement in time. The sequence begins to the far right (at the 'o') and ends on the far left of the diagram (at the 'x').

MDS yields a representation within which the arrangement of points depends on the similarities among the patterns to which they correspond. Thus, points close together in the diagram represent similar patterns of activation across the model's fifty hidden units. For example, consider the steps in the sequence for which the action is pour (indicated by diamond characters in Figure 12). There are three such steps, one representing pouring of the grounds, another pouring from the sugar packet, and the third pouring from the carton. These are located close together in the figure, reflecting the fact that the network uses similar internal representations during these three pouring events. In contrast, consider the step where the action peel-open is performed on the carton (plus sign). Since this action occurs only once in the sequence, and in combination with an input pattern that is also unique, the internal representation corresponding to this step is quite distinctive, appearing in the figure some distance from its nearest neighbor.

An interesting case is provided by the two points labeled with asterisks. The two steps represented here involve the same action, stir. They also involve precisely the same per-

ceptual input: (fixated: cup, 1-handle, brown-liquid, held: spoon; held: spoon). The internal representations associated with the two steps are similar but not identical, as shown by their non-overlapping positions in the diagram. The distance between them reflects the fact that the two stir actions are performed in different temporal contexts. One is performed just after the coffee grounds have been added, the other just after adding sugar. This difference in context is signaled by the pattern of activation conveyed by the network's recurrent connections, as described in the Introduction. The input the hidden layer receives via these connections allows it to assume different patterns of activation for the same stimulus-response mapping when this mapping is performed in different temporal contexts.

This case provides a first example of how the network's internal representations capture the hierarchical structure of the tasks on which it has been trained. Specifically, it illustrates how the network simultaneously encodes information at two levels of temporal structure: the level of individual actions and the subtask level. The fact that the two points involve the same action is reflected in their proximity to one another, relative to the other points in the figure. The information that they occur as part of two different subtasks, grounds- and sugar-adding, is reflected by the existence of the small but non-zero distance between the points (see also Elman, 1991, 1993).

The network's internal representations also reflect information at a broader level of temporal structure, that of the overall task being performed. For illustration, consider once again the sugar-adding task as performed in two different task contexts, tea-making and coffee-making. Figure 11 showed the network's overt performance in these two contexts. Figure 13 shows an MDS plot of the internal representations arising during the same steps. The trajectory shown with solid lines represents sugar-adding as performed as part of the coffee task. The dashed trajectory represents sugar-adding during tea-making. The first point to note is that the corresponding points in the two trajectories are located near to one another. As in the previous example, this reflects the fact that the corresponding steps involve precisely the same environmental input patterns and actions. Once again, however, steps involving identical actions are represented slightly differently, resulting in a shifting of the entire trajectory. The similarity between the two trajectories reflects the fact that they represent the same subtask; the positional shift between them reflects their embedding in two different tasks.

The hierarchical organization evident in the model's hidden representations reflects far more than a passive encoding of task structure. In fact, it is the key to how the network executes hierarchically structured sequences. As we have emphasized, performance in hierarchical domains—including the present tasks—requires a memory mechanism capable of preserving contextual information. The internal representations diagramed in Figures 12 and 13 show such a mechanism in action. In Figure 12, the two patterns representing the stir action preserve information about the subtask context in which that action is performed, allowing the network to move on to the appropriate next subtask once the action is

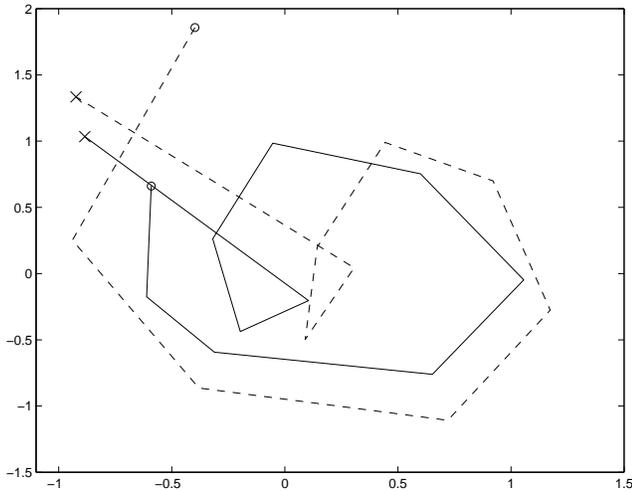


Figure 13. MDS plot of the hidden representations arising during the sugar-packet sequence as performed in the context of coffee-making (solid) and tea-making (dashed).

completed. Analogously, the fact that the internal representations diagrammed in Figure 13 preserve information about the overall task context allows the network to select the correct subtask (cream-adding or drinking) once sugar-adding is complete, as seen in Figure 11.

The same memory mechanism underlies the network's success in avoiding repetitions and premature selection of actions. Consider once again the network's situation at the end of the cream-adding subtask. Here, as discussed above, the correct action is either put-down (beginning the drinking subtask) or fixate-sugar, the correct choice depending on whether or not sugar has already been added. Choosing between these incorrectly would mean making either an error of omission or a repetition error. Because the coffee in the cup looks the same whether it contains sugar or not, the network must keep track of whether the sugar sequence has been completed. This information is encoded—just like information about the overall task context—in the network's internal representations. Figure 14 shows an MDS plot of the steps in the cream sequence. The solid trajectory reflects the steps as performed prior to sugar-adding, the dashed trajectory after. Once again, although the corresponding steps from each sequence involve identical environmental inputs and identical actions, they are represented slightly differently, reflecting the preservation of contextual information. At the end of the cream sequence, it is this information that allows the network to enter the correct subtask, avoiding a repetition or omission of sugar-adding.

An important aspect of the internal representations we have been discussing is that, rather than being built into the model from the start, they are learned through exposure to the task domain. Interestingly, a close look at how the model's internal representations develop over the course of learning reveals that this is influenced by the hierarchical structure

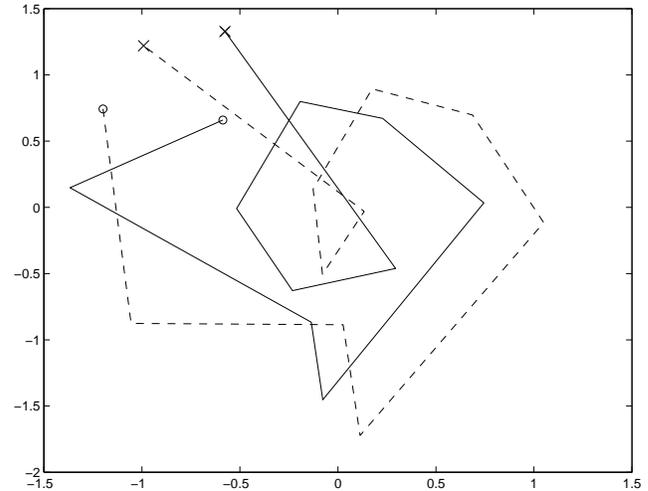


Figure 14. MDS representation of the hidden representations arising during performance of the cream-adding sequence, as performed before (solid) and after (dashed) sugar-adding.

of the tasks being learned. Figure 15 shows another MDS plot of the cream-adding sequence. As in Figure 14, this figure shows the sequence as it occurs in two contexts, before and after sugar-adding. The three panels of the figure show the relevant internal representations at three stages of training. The data in the left panel were collected after 100 passes through the training set. Here, only small differences are evident among the hidden representations for the various steps in the sequence. The picture has changed substantially by 1,000 epochs of training, as shown in the figure's center panel. Here, the steps in the sequence have diverged, indicating that the network has learned to represent differently the stimulus-response mapping occurring on the various steps. Note, however, that the network follows nearly the same trajectory, regardless of the task context. The network has not yet learned to encode contextual information above the subtask level. By epoch 10,000 (right panel), the network has learned to encode this information, yielding a pattern analogous to that shown in Figure 14 (the differences between the two figures derive from the fact that the data in Figure 15 were collected during a separate training run). This sequence of events in the network's development underlies the sequence seen in Figure 10, which showed the evolution of the model's overt performance. Once again, the course of learning recapitulates the hierarchical structure of the tasks being learned. First, there is learning of lower levels of temporal structure, then successive acquisition of more global levels (see also McClelland, McNaughton, & O'Reilly, 1995; Rogers & McClelland, submitted; Servan-Schreiber et al., 1988, 1991).

#### *Performance on a Quasi-Hierarchical Task.*

One characteristic of normal routine action that we have emphasized is its context dependence. This refers to the way in which the details of a task or subtask vary depending on

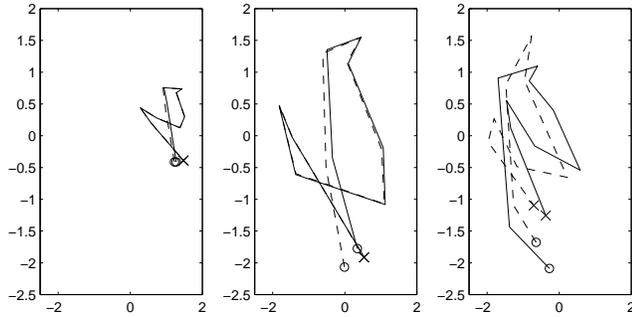


Figure 15. MDS plot of the hidden representations arising during cream-adding, as performed before (solid) and after (dashed) adding sugar. Left: After 100 epochs of training. Center: 1,000 epochs. Right: 10,000 epochs.

the particular situation in which it is performed. In our basic implementation of the coffee task, such context dependence is largely filtered out. The steps followed in adding cream, adding sugar, drinking, etc. are the same regardless of the temporal context in which they occur. In order to address the issue of context dependence, we therefore carried out a separate simulation. Here, the model used in the central simulation was retrained using a set of tasks intended to implement the example given in the Introduction, involving a waiter preparing coffee for three regular customers, each of whom expects a different amount of sugar.

Three new instruction units were added to the model, corresponding to requests for coffee with no sugar, with one spoonful of sugar, and with two spoonfuls, respectively. The model was trained in the same manner as in the main simulation, but using a training set containing modified versions of the coffee sequence shown in Table 1. Each sequence began with the initial input used in that sequence, but now accompanied by one of the new instruction units. The sequence for making coffee without sugar followed the original coffee sequence, but omitted the entire sugar sequence. The sequence for making coffee with one spoonful of sugar was drawn directly from the original set of four coffee sequences (GROUNDS  $\rightarrow$  SUGAR(BOWL)  $\rightarrow$  CREAM  $\rightarrow$  DRINK). The sequence for making coffee with two spoonfuls of sugar was identical to the previous sequence, but the sequence fixate-sugarbowl  $\rightarrow$  scoop  $\rightarrow$  fixate-cup  $\rightarrow$  pour appeared twice in succession, rather than being performed once as in the original version of the task. In view of the fact that no variability in performance was required of the model, the hidden layer was initialized by setting all unit activations to 0.5, rather than to random values. Training was terminated when it was evident that learning had reached a final plateau (20,000 epochs).

Testing of the fully trained model indicated that it was able to perform all three versions of coffee-making accurately. In accordance with the instruction unit activated on the first cycle of processing, the network reproduced the appropriate version of the coffee-making sequence from the training set, either omitting sugar or adding one or two spoonfuls from

the sugar bowl.

The model's method of dealing with the sequences involved in this simulation provides a contrast to traditional, hierarchically structured models of sequential action. As discussed in the Introduction, such models face an uncomfortable choice between representing different variations of a sequence using a single unit or using multiple independent units. The present model, in contrast, provides a natural means for simultaneously encoding the relatedness of two sequences while at the same time representing their differences. Once again, the point is made clear by an examination of the internal representations the model uses during task processing. Figure 16 shows a set of MDS plots based on the patterns of activation arising in the model's hidden layer on each step of processing in each of the sequences in the training set. The sequence in which no sugar was added is shown in the left column, the sequence with one spoonful of sugar in the middle column, and the sequence with two spoonfuls of sugar in the right column. The steps involved in the sugar-adding sequence itself are shown in the middle row of panels. The sequence of steps preceding sugar-adding are shown in the top row, and those following it in the bottom row. What the figure indicates is that there is a family resemblance among the internal representations the model uses across the three versions of the task. Within each row of the figure, the representations trace out roughly similar trajectories, reflecting the fact that the model has picked up on the similarity among the sequences the tasks involve. The trajectories are not identical, however. It is the small differences among them that enable the network to produce differential behavior at the appropriate junctures.

The patterns pertaining to the two versions of sugar-adding indicate that the network has capitalized on the fact that there is shared structure between the two sequences; where there are steps shared by the two versions of sugar-adding, very similar internal representations are used for each. Similarly, since the grounds-adding, cream-adding, and drinking sequences are identical across the three versions of the coffee-making task, the network uses very similar internal representations in executing these sequences. These internal representations are not totally insensitive to overall task context, however. The representations arising during the grounds-adding sequence, in particular, differ slightly depending on which version of coffee-making is being undertaken. This reflects the fact that the network must preserve information about which customer is being served in order to behave correctly at the end of grounds-adding.

In learning to perform the waiter's task, as in learning to perform the basic coffee task itself, the network must strike an important balance. For generalization, it must capture the similarities or identities between subtask sequences. On the other hand, in order to handle the idiosyncratic details of its task, and in order to perform correctly at so-called decision points, it must also capture important differences in context. Through learning, the network discovers internal representations that effectively balance these two constraints, simultaneously capturing similarities and differences between familiar action sequences. As shown in the next section of the

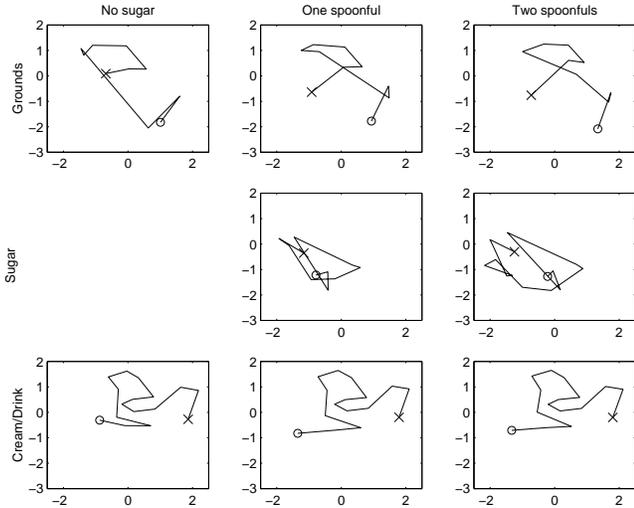


Figure 16. MDS plots of the internal representations underlying the model’s performance in the waiter scenario.

simulation, the solutions it finds also make it prone to certain kinds of errors.

**Results: Slips of Action**

In this portion of the study we asked whether weakly taxing the mechanism responsible for preserving context information, by adding low levels of noise, would lead to errors resembling the slips of action made by normal subjects. Not surprisingly, errors became more frequent under increasing noise (see Figure 18). Since slips of action in normal subjects are associated with levels of performance where overall error rates are low, this section of the study focused on model performance at noise levels associated with relatively infrequent errors—noise with variance in the range 0.02 to 0.1.

In this range, in keeping with human behavior, the model’s errors tended strongly to occur at “decision points,” corresponding here to the transitions between subtasks. One way to illustrate this behavior is with a survival plot, as shown in Figure 17. The data plotted here are based on 100 test runs on the coffee-making task, applying noise with variance 0.1. The horizontal axis indexes the steps in the task. The vertical axis shows the number of trials for which no error had yet occurred at the corresponding step. Occurrence of errors at a given point is indicated by a sudden step down in the diagram, the size of which reflects the frequency of errors at that step. The plot contains large drops at steps number 11, 22, and 33, the three points in the task where a subtask has just ended and a new one begins. Thus, once the model has entered into a subtask like sugar- or cream-adding, it usually completes that subtask correctly. Errors occur almost exclusively at the transitions between subtasks.

Also in keeping with empirical findings, the model’s errors tended to take the form of recognizable subtask sequences, inserted at the wrong moment but nonetheless drawn intact from somewhere in the training corpus. In some

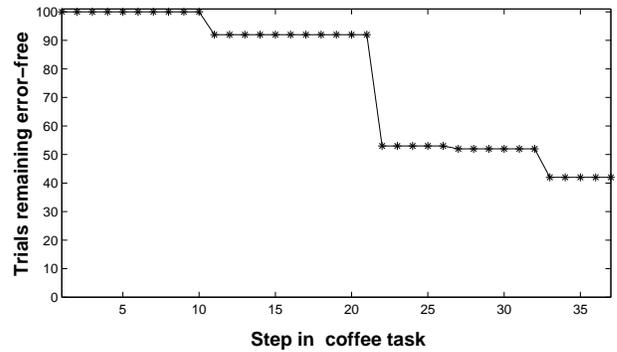


Figure 17. Survival plot for the coffee task under noise with variance 0.1. On the x-axis are steps in the sequence. The y-axis indicates the number of trials, from a total of 100, that remained error-free at the corresponding step. Data is collapsed across the four versions of the task.

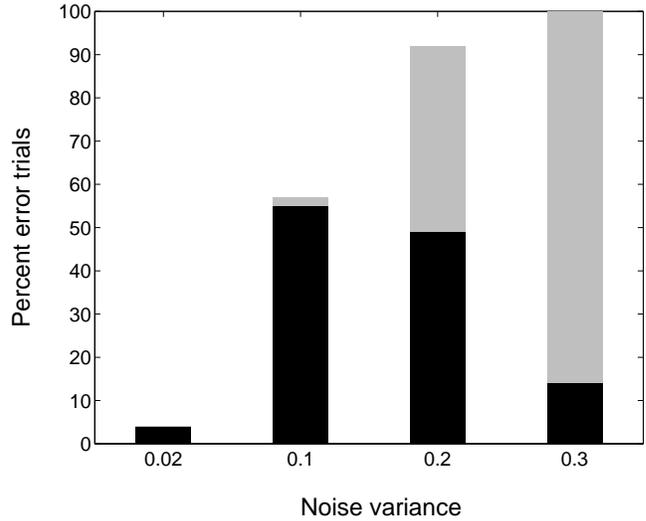


Figure 18. Number of trials in 100 containing at least one error, at four levels of noise. The black portion of each bar indicates trials that contained only errors involving intact but displaced subtasks. The gray area indicates trials involving within-subtask disorganization.

instances the sequence inserted came from earlier within the task being performed, resulting in a perseveration or repetition error (e.g., adding sugar twice, either in succession or with intervening subtasks). In other instances the inserted sequence came from later in the task, resulting in a subtask omission (e.g., leaving out the sugar subtask and skipping directly to cream-adding and then drinking). In still other cases the inserted subtask came from outside the task being performed, resulting in an intrusion error (e.g., adding cream during tea-making).

This tendency for errors to take the form of intact but displaced subtask sequences is illustrated in Figure 18. The

black portion of each bar in this diagram indicates the number of trials at each level of noise that contained *only* errors involving subtask displacement. The gray portion stacked above this shows the number of trials containing at least one error involving a disordered subtask sequence (e.g., fixate-carton → pick-up → peel-open → fixate-cup → put-down). As the figure makes clear, at low overall error rates, errors primarily took the form of intact subtask sequences.

In summary, the errors produced by the model under low levels of noise displayed three principal characteristics of slips of action produced by normal subjects: The errors tended to occur at branch points; they tended to take the form of displaced but well-formed action sequences; and they involved omissions, repetitions, or intrusions.

#### Analyses.

In order to understand these characteristics of the model's errors, it is useful first to consider the basic mechanisms by which errors occur. Recall that, in the present simulations, noise was added to each hidden unit's activation at the end of every processing cycle, following action selection. The effect of this is to distort the information being transmitted over the recurrent connections within the hidden layer. As indicated by the analyses presented in the last section, the normal role of this information is to provide a representation of temporal context, to be combined with environmental information in selecting the next action. Thus, noise acts to distort the model's representation of temporal context.

The functional consequences of such distortion follow from a basic property of connectionist models: If faced with a novel or distorted representation (either an external input or, as here, an internal representation), such models tend to respond based on that representation's *similarity* to more familiar ones (Rumelhart et al., 1996). When the present model is faced with a distorted context representation, there are two possible outcomes. If the distorted context representation resembles the undistorted version more closely than any other context representation the model has dealt with during learning, the correct action will still be produced. An error occurs when the distorted context representation happens to resemble a pattern the network has learned to associate with another action.

For illustration, consider the following relatively common error: following adding grounds, sugar from the sugar bowl, and cream, the model selects fixate-sugar and enters into a second round of sugar-adding, committing a perseveration error. Ordinarily, in order to select the correct action following cream-adding (putting down the spoon, in preparation to drink), the model relies on a context representation that carries the information that sugar has already been added (see Figure 14). However, in the case of this perseveration error, noise has had the effect of distorting the context representation so that it resembles the one the model normally uses to represent the fact that sugar has *not* yet been added. In response, the model produces the action appropriate to that context, fixate-sugar.

Evidence for this explanation is provided by the data shown in Figure 19. This focuses on the context representa-

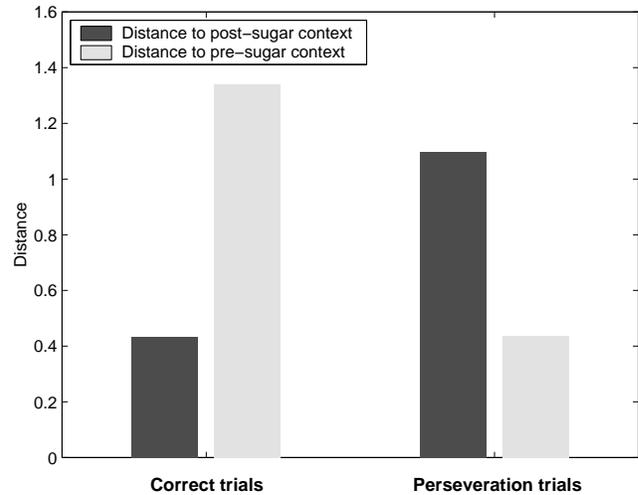


Figure 19. Comparison of the internal representations arising on the last step of the cream subtask, for trials where this step led into a sugar perseveration error and for correct trials. Data for both correct and error trials are based on averages across a sample of size 10.

tions arising at the end of the cream-adding sequence, comparing these between trials where the model correctly headed into the drinking sequence and trials where a sugar perseveration occurred. For each trial-type, the plot shows the average distance of the actual context representation from two reference patterns, both produced by the model when run without noise: (1) the context representation normally arising at the end of cream-adding when sugar *has* already been added (the “post-sugar” context), and (2) the one arising at the end of cream-adding when sugar has *not* yet been added (“pre-sugar”). What the figure indicates is that the context representations on correct trials, though distorted, tended to resemble the usual post-sugar pattern more closely than the pre-sugar pattern. In contrast, on trials where a sugar perseveration occurred, the context representations present at the point of the error tended to resemble the pre-sugar pattern more closely. On these trials, noise has in effect caused the model to “forget” that it previously added sugar, by producing a context pattern resembling the one ordinarily used to represent the situation where sugar has in fact not been added.

Importantly, the effects of such context confusions are most likely to show up at branch points. This is because branch points involve a situation where *similar* context representations are associated with *different* actions. The idea can be illustrated using Figure 14. The final points in the two traces shown here correspond to the pre- and post-sugar context representations arising at the end of cream-adding. These points are located near to one another, indicating that the corresponding context patterns resemble one another closely. As a result of this resemblance, it is particularly easy for a slight distortion in one context representation to produce a pattern resembling the other. Since the two pat-

terns are associated with different actions, this slight distortion leads to a branch-point error.

Note that while Figure 14 shows that the two contexts associated with the end of the cream-adding sequence are similar to one another, it also indicates that steps within the sub-task sequence have nearby neighbors, as well. For each step there are two relevant contexts, one indicating the pre-sugar situation, the other the post-sugar situation, and the members of most of these pairs lie near to one another. Since such neighboring relations are part of the situation that leads to errors at branch points, an obvious question is why they don't tend to lead to errors away from such junctures, within the boundaries of subtask sequences. The answer becomes clear when one considers what the consequence would be of distorting any of these context representations so that it came to resemble its counterpart. For illustration, consider the step where the correct action is to pour cream from the carton into the cup. Suppose again that sugar has already been added, but that noise has distorted the context representation so that it resembles the pattern usually present on the same step when performed prior to adding sugar. Once again, it is as if the network has forgotten that sugar has been added. However, this does not lead to an immediate error, because on this particular step the correct action is pour whether or not sugar has been previously added. The same distortion that leads to an error at the branch point does not cause an error prior to that point because the particular piece of information that has been corrupted is not yet relevant to action selection.

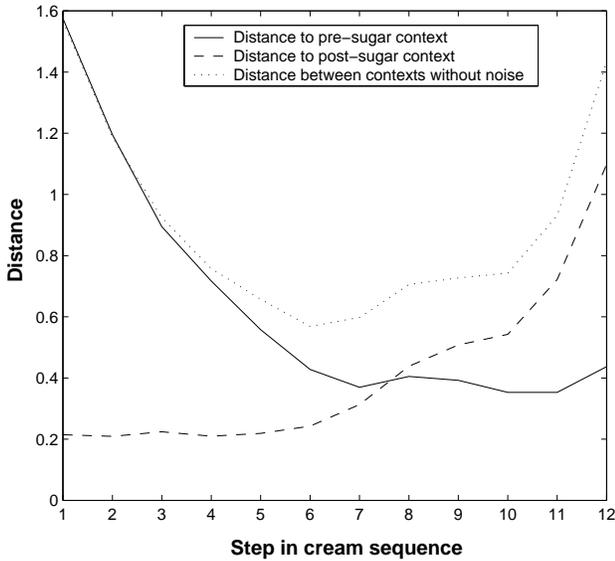
Interestingly, although context distortion away from a branch point may not cause an immediate error, it may contribute to the likelihood that an error will occur at an upcoming branch point. This is because once noise has caused the network to misrepresent its temporal context, this misrepresentation will propagate to subsequent cycles of processing. Resuming the scenario above, the context representations arising during the remaining steps in the cream sequence will resemble those associated with the corresponding steps when performed prior to adding sugar. Once again, because each of these steps is performed identically regardless of whether sugar has already been added, the ongoing misrepresentation of context causes no immediate error in action selection. The situation changes, however, once the network arrives at the branch point lying at the end of the cream-adding subtask. Because here the pre- and post-sugar contexts are associated with different actions, the distortion in the context representation will lead to an error.

This analysis suggests the possibility that the model's branch-point errors may often result from context distortions originating not at the branch point itself, but at earlier points in time. Indeed, in principle the distortion need not arise abruptly on any given step, but may emerge gradually due to the accumulation of small distortions over a number of steps preceding the branch point. An analysis of the model's internal representations confirms that this is in fact the case; at low levels of noise, branch-point errors involve a gradual drift in the representation of context beginning several steps before the branch point, but becoming manifest at the level

of action selection only once the branch point is reached. For illustration, we focus once again on the example of the sugar perseveration, where the model selects fixate-sugar after having just added coffee, sugar, and cream. Figure 20 represents an analysis of the context representations arising on each step in the cream sequence, leading up to this slip. Like the context representations examined in the previous diagram, here the representation on each step is visualized in terms of its distance from two reference vectors, produced by the model on the corresponding step in the absence of noise. The dashed line shows the distance of the actual context representation from the pre-sugar reference pattern (produced on a run where the model added cream prior to sugar). The solid line shows the distance of the same representation from the post-sugar reference.

The dashed line in the figure curves upward. This indicates that, over the steps leading up to a sugar perseveration error, the activations making up the context representation tend to drift gradually away from their canonical post-sugar values; the network gradually forgets that sugar has already been added. The solid line, showing the distance of the context representations from their pre-sugar references, falls over the course of the cream sequence. Again, this shows the context representations gradually losing the information that sugar has already been added to the cup. Eventually, the solid and dashed lines cross, representing the point at which the network has effectively forgotten that sugar has been added. Note that, on average, this cross-over occurs several steps prior to the step on which the error actually occurs. The network's representation of context moves gradually from the pre-sugar to the post-sugar pattern over the course of the cream subtask, but the distortion does not lead to an error until the end of the sequence, the first point at which action selection depends on the information that has been lost.

One reason that branch-point errors are usually associated with a gradual representational drift beginning several steps earlier is that, at low levels of noise, only small distortions occur on each step. It thus requires several incremental distortions to sufficiently disrupt the system's function. However, a further examination of the model's context representations indicates that there is also another reason, namely that it is easier for the model to confuse the present context with another one when processing is toward the middle of a sub-task sequence than when it is near the beginning or end of one. To explain why this is so, we return to the data diagrammed in Figure 20. Recall that the distance data in this figure were computed using two sets of reference patterns, the context representations occurring in the pre- and post-sugar situations, collected when the network was run without noise. While the context representations occurring under noise drift toward and away from these reference patterns, it is interesting to note that the reference patterns themselves vary in their distance from one another over the course of the cream-adding sequence. Specifically, as illustrated by the dotted line in Figure 20, the patterns from corresponding steps approach one another in the first half of the sequence and then move apart toward the end. The initial reduction in

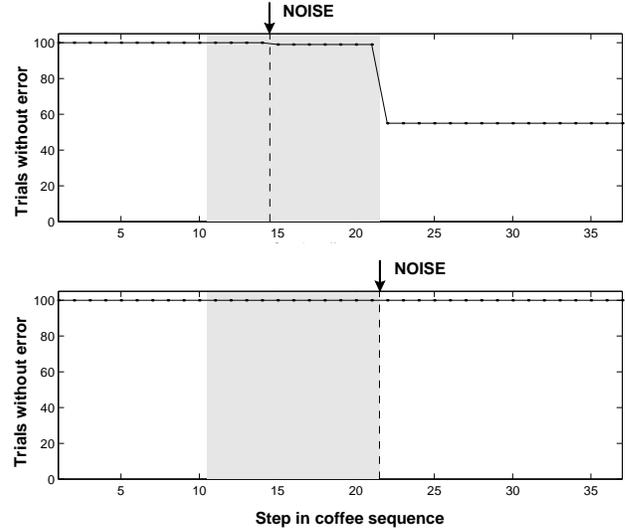


*Figure 20.* The steps of the cream-adding sequence, leading up to a sugar perseveration error. Solid trace: distance of the average context pattern from the pre-sugar reference for the corresponding step. Dashed: distance from the post-sugar reference. Dotted: distance between the two reference vectors. Data based on averages across a sample of size 10.

distance reflects the fact that, with each new step, the temporal context represented by the two becomes more similar. The separation toward the end reflects an “anticipation” of the need to make the context representations as different as possible at the branch point, in order to support differential behavior.

The pattern illustrated in the figure means, in effect, that the network represents the pre- and post-sugar situations more similarly toward the center of the subtask sequence than near the branch points at either end. As a result, the center of the subtask represents a point in processing where contextual information is particularly vulnerable to the effects of noise. It is particularly easy here for noise to alter a post-sugar context pattern so that it resembles the corresponding pre-sugar pattern, or vice versa. At the beginning and end of the subtask, where the standard context patterns are more distinct, noise is less likely to have this effect.

The increased vulnerability to noise during the middle of the subtask can be demonstrated by examining the impact of injecting noise only on a single step of processing. If noise is added on a step near the middle of a subtask sequence, this is more likely to lead to an error than if the same amount of noise is added on a step toward the end of the subtask. This effect is illustrated in Figure 21, which shows survival plots for two sets of 100 test runs of the model, all of which began with the grounds-adding sequence followed by cream-adding. In one set of trials, noise with variance 0.5 was added to the context representation just prior to the fifth step of the



*Figure 21.* Survival plot showing the effect of injecting noise on a single step during the cream subtask. Dashed lines indicate steps on which noise was injected. Steps belonging to the cream-adding subtask are set off in gray. Top: injection of noise before step 15. Bottom: injection of noise before step 22.

cream sequence (correct action: *fixate-cup*). In the other set, noise was added just prior to the branch point at the end of the sequence (correct action: *fixate-sugar*). As the figure shows, noise injected at the branch point produced no errors at all. In contrast, noise of the same magnitude injected toward the center of the subtask sequence resulted in errors on nearly half of all trials. Note that the vast majority of these errors did not occur immediately. In line with the above discussion, disruption to contextual information within a subtask sequence did not affect action selection until a branch point was reached. The new point the figure illustrates, however, is that a distortion of contextual information is most likely to affect overt behavior when it occurs toward the middle of a subtask sequence. This aspect of the model leads directly to testable predictions about human performance, a point to which we return in the General Discussion.

To this point, we have focused on the factors that determine when errors occur. An equally important characteristic of the model’s errors is that, wherever they occur, they tend to take the form of misplaced subtask sequences, drawn intact from somewhere in the training set. As it turns out, this aspect of the model’s behavior stems from the same factors we have already been discussing. The same misrepresentation of context that leads the network to take a first misstep is also responsible for its tendency to continue on to produce an entire out-of-place subtask. For illustration, consider once again the case of the sugar perseveration error. As detailed above, the network’s first mistake, selection of the *fixate-sugar* action, occurs because the context representation has been distorted to resemble the one normally present at the beginning of the sugar sequence. Naturally, because the net-

work represents itself to be in that situation, it produces not only the first step of the sugar subtask, but the entire sugar sequence.

#### *Effect of Relative Task Frequency.*

One particularly interesting aspect of the empirical data concerning intrusions is that these errors show an effect of relative task frequency; intrusions tend to involve a lapse from a less frequently executed task into a more frequent one. We addressed this phenomenon in a separate simulation, in which the model was retrained three times using different relative frequencies of coffee- and tea-making. The first set included five times as many instances of coffee-making as tea-making, the second equal proportions of the two tasks, and the third five times as many instances of tea-making. The total number of target sequences (coffee plus tea) was balanced across sets. Each training set included the same group of background examples that appeared in the original training set. Training was terminated after it was evident that a final plateau in error had been reached (5000 epochs). Following training, each version of the model was tested in the standard fashion, using the *instruct-tea* unit on the first cycle of processing and noise with a variance of 0.1.

Evaluation of the model's performance in these three contexts focused on the frequency of lapses from tea- into coffee-making, indicated by the error of adding cream to tea. Specifically, we asked whether the frequency of this error would vary inversely with the relative frequency of tea-making during training. In accordance with empirical data, the model's behavior did show an effect of task frequency on the tendency to lapse from one task into another. Figure 22 shows the frequency of the cream-into-tea error following training on the three example sets. The lapse error occurred more frequently as the tea task became less frequent in training.<sup>6</sup>

The mechanism behind this behavior can be understood in much the same terms we have just been discussing. Like the errors we have already considered, the cream-into-tea error begins when distortion to the context representation leads it to resemble another pattern with which the network is familiar, but which is associated with a different action. On trials where the network performs correctly, the context representation at the end of the sugar sequence reflects that fact that the overall sequence began with steeping the tea, and thus that the current task is tea-making. In the presence of this context representation, which we will refer to simply as the *tea* pattern, the network selects the correct action, *put-down* (in preparation for drinking). In contrast, on trials where the cream-into-tea error occurs, noise has had the effect of causing the context representation to resemble the pattern normally arising at the analogous point (the end of sugar-adding) in the coffee task. Because this *coffee* pattern is associated with the action *fixate-carton*, that action is selected, and the cream-into-tea error begins.

In order for the cream-into-tea error to occur, then, noise must move the context representation away from the tea pattern and toward the coffee pattern. Figure 23 (top) illustrates how this movement affects action selection. The data shown here were produced by applying, and holding constant, the

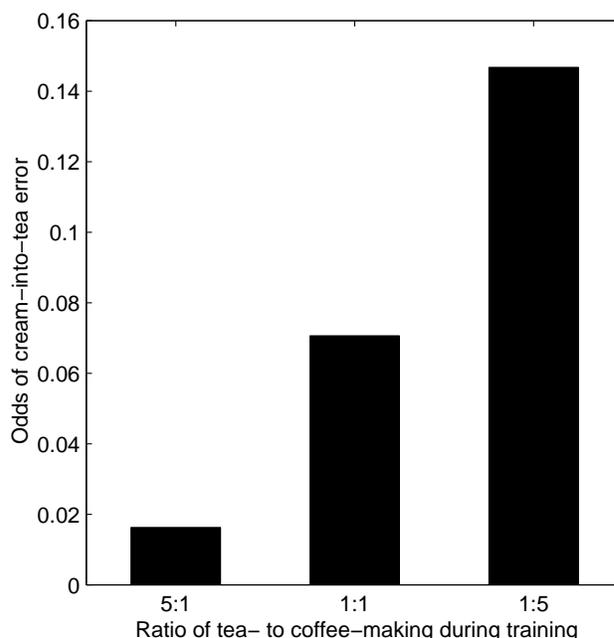


Figure 22. Odds of the cream-into-tea lapse, following training involving varying proportions of coffee- and tea-making. Each data point is based on 500 trials, using a noise level of 0.1.

environmental input normally present at the end of the sugar-packet sequence in the tea task, while applying a series of gradually varying context representations. These were produced by starting with the tea pattern and gradually distorting it in the direction of the coffee pattern. The effects of this gradual distortion are plotted from left to right in the figure. The data themselves relate to the activation of two output units, *put-down*, the correct action in the tea context, and *fixate-carton*, the correct action in the coffee context. As the pattern becomes less similar to the tea context, the network activates the *put-down* action less strongly, and as the pattern comes to resemble the coffee context more, the network more strongly activates *fixate-carton*. The point at which the two traces in the figure intersect provides an indication of the distance the context pattern must travel before the cream-into-tea error will occur. As illustrated in the center and bottom panels of Figure 23, variations in relative task frequency influence where this cross-over occurs along the path from the tea pattern to the coffee pattern. When tea-making occurs more frequently during training, the cross-over point lies further from the undistorted tea context, meaning that a more severe distortion is required to produce the cream-into-tea error. As a result, the error occurs less frequently. Conversely, when tea-making is relatively infrequent during training, the

<sup>6</sup> Further simulations reproduced the effect when the absolute number of presentations of the tea task during training was held constant across training sets, indicating that the increase in the error rate cannot be attributed simply to inexperience with tea-making alone.

cross-over point lies closer to the tea-making pattern. This means that a smaller distortion will cause the error, explaining why it occurs more frequently (see also Rogers & McClelland, submitted).

### Results: Action disorganization syndrome

Mild degradation of the model's representation of temporal context led to errors resembling everyday slips of action. The next segment of the study tested the prediction that more severe degradation would lead to behavior resembling that of patients with ADS. Specifically, it was asked whether higher levels of noise would: (1) produce increasing fragmentation in sequential structure, as measured by the number of independent actions (see Figure 7), (2) produce examples of the specific error types observed in ADS, and (3) reproduce the relationship between the proportion of omissions and overall error rate reported by Schwartz et al. (1998; see Figure 8).

In keeping with the performance of ADS patients, the sequences produced by the model did become increasingly fragmented with increasing noise. At noise levels above 0.1 errors first began to appear within subtask boundaries, rather than only at the transitions between subtasks (see Figure 18). A typical example is shown in Table 5 (left). With increasing noise, sequencing both within and between subtasks became increasingly disrupted (Table 5, center). At high levels of noise, only short fragments of the original subtask sequences could be discerned, as shown in Table 5 (right). At extreme levels of noise, trials were increasingly taken up with extended periods of rather aimless "toying" behavior, which is also characteristic of the behavior of highly impaired ADS patients (Schwartz et al., 1991).

A rough quantification of the degree to which sequential structure is disrupted is provided by the frequency of independent actions (as defined by Schwartz et al., 1991 and the Appendix). As shown in Figure 24, the proportion of independent actions increased smoothly with the severity of noise. It is significant that the change in fragmentation is graded, since the empirical data show that graded changes in the fragmentation of patient performance can occur over the course of recovery (Schwartz et al., 1991, and Figure 7 above).

As the data in Table 5 and Figure 24 make clear, the occurrence of an error did not throw the model completely "off track." Following an error, the model typically fell into behavior bearing some resemblance to the sequences presented during training. This tendency reflects the attractor dynamics that are characteristic of sequential networks (Gupta & Dell, 1999; Jordan, 1994; Perlmutter, 1989). When placed in a state different from those occupied in executing sequences learned during training, recurrent models have a tendency to get drawn back into familiar lines of behavior over ensuing steps. The behavior of the present model under noise can be understood as reflecting a balance between this corrective tendency and the direct effects of noise; noise acts to knock the model out of learned patterns of behavior, and the model's attractor dynamics tend to draw it back into those patterns. With increasing noise, the former process increas-

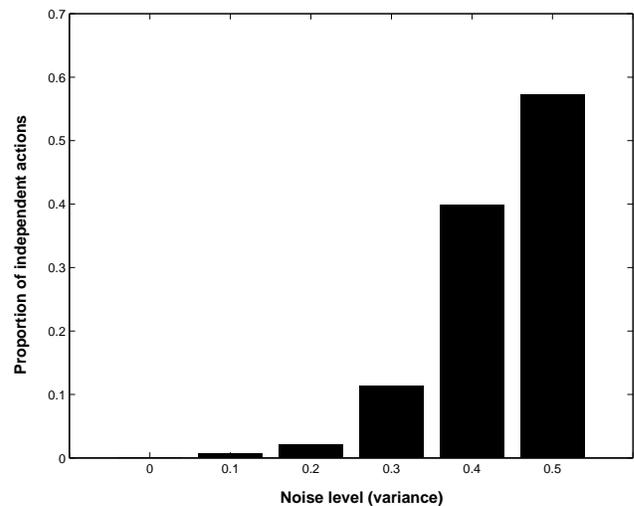


Figure 24. Proportion of independent actions produced by the model at several levels of noise severity.

ingly dominates over the latter, and increasingly fragmented behavior is observed.

Piecemeal examination of the model's errors revealed instances of each of the error-types described by Schwartz et al. (1998) as occurring in ADS. As shown in Table 6, the majority of errors were either omission or sequence errors. However, examples of object and gesture substitutions, action additions, tool omissions, and quality errors also occurred. As in ADS, the most frequent error type was omission. Such errors included omissions of both entire subtasks (usually the cream and/or sugar sequences) and fragments of subtasks. As the frequency of errors grew with increasing noise, the proportion of omission errors rose more rapidly than the proportions of other errors, as reported for ADS patients (Schwartz et al., 1998). Figure 25 shows the number of omission errors along with the number of sequence errors occurring at a range of noise levels. With increasing noise the number of omission errors grew steeply, while the number of sequence errors grew very little (compare Figure 8). This finding is particularly significant, given the fact that a recent hierarchical model of ADS (Cooper & Shallice, 2000) did not reproduce the relationship between error-rate and the proportion of omission errors described by Schwartz et al. (1998).

While the aim of the simulation was to address qualitative rather than quantitative patterns of error, for later reference we take note of the relatively low incidence of substitution and addition errors in the model, as compared with their incidence in ADS. We believe the rarity of these errors may be attributed largely to implementational issues, which we take up in the General Discussion.

In summary, the behavior of the model reproduced several core characteristics of behavior in ADS: Deterioration in per-

Table 5  
*Examples of the Model's Performance Under Increasing Noise*

pick-up coffee-pack	pick-up coffee-pack	pick-up cup
pull-open coffee-pack	pull-open coffee-pack	sip
pour coffee-pack into cup	put-down coffee-pack	put-down cup
put-down coffee-pack	pick-up coffee-pack	pick-up carton
pick-up spoon	pour coffee-pack into cup	peel-open carton
stir cup	put-down coffee-pack	put-down carton
put-down spoon	pick-up spoon	pull-off sugar bowl lid
pull-off sugar bowl lid	stir cup	put-down lid
put-down lid	put-down spoon	pick-up spoon
pick-up spoon	pick-up sugar-pack	put-down spoon
stir cup	tear sugar-pack	pick-up coffee-pack
put-down spoon	pour sugar-pack into cup	put-down coffee-pack
pick-up carton	put-down sugar-pack	pick-up sugar bowl
peel-open carton	pick-up cup	put-down sugar bowl
pour carton into cup	put-down cup	pick-up coffee-pack
put-down carton	pull-off sugar bowl lid	pull-open coffee-pack
pick-up spoon	put-down lid	pour coffee-pack into cup
stir cup	pick-up spoon	put-down coffee-pack
put-down spoon	scoop sugar bowl with spoon	pick-up spoon
pick-up cup	put-down spoon	put-down spoon
sip cup	pick-up cup	pick-up cup
sip cup	sip cup	sip cup
say-done	sip cup	say-done
	say-done	

*Note:* Fixate actions are omitted. Left: This sequence skips from opening the sugar bowl directly to stirring, but is otherwise correct. Center: Among other errors, this sequence omits cream-adding and performs the sugar subtask twice (omitting steps in both cases). Right: Fragmented behavior under a high level of noise (variance 0.4).

Table 6  
*Examples of Individual Error-Types*

Type	Example	Percentage
Omission	Sugar not added	77
Sequence		15
Anticipation	Pour cream without opening	
Perseveration	Add cream, add sugar, add cream again	
Reversal	Stir water then add grounds	
Other		8
Obj. substitution	Stir with coffee packet	
Gesture substitution	Pour gesture substituted for stir	
Tool omission	Pour sugar bowl into cup	
Action addition	Scoop sugar with, then put down, sugar bowl lid	
Quality	Pour cream four times in a row	

*Note:* Frequencies are based on a sample of 100 trials under noise with variance 0.2. Examples are drawn from sequences produced under a variety of noise levels.

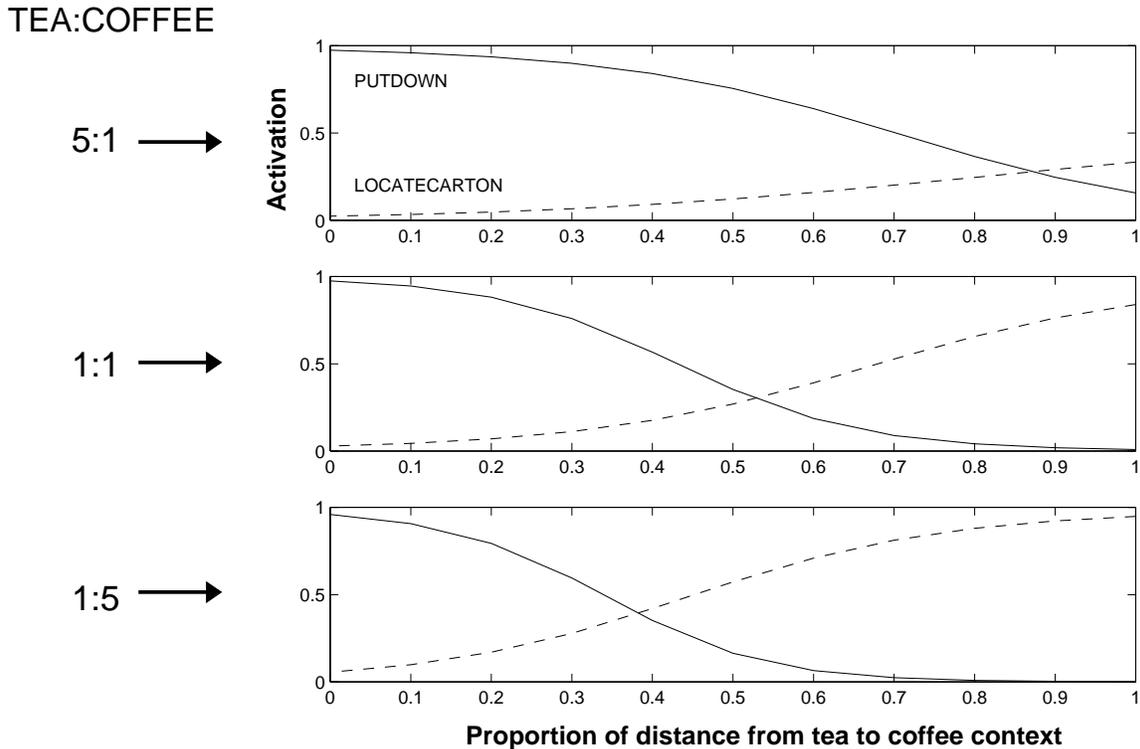


Figure 23. Effect on action selection of progressively distorting the context representation at the end of the sugar sequence in the tea task toward the corresponding coffee context. Ratios of tea to coffee during training: top, 5:1; middle, 1:1; bottom, 1:5.

formance manifested as a gradually increasing fragmentation of sequential structure, a specific set of error-types occurred, and the proportion of omission errors increased with overall error rate.

#### Analyses.

Perhaps the clearest difference between the model's behavior at low and high levels of noise is that, at low noise levels, errors occurred primarily at the branch points between subtasks, while at higher levels of noise errors began to occur within subtask boundaries (see Figure 18). Despite the importance of this distinction, a close look at non-branch-point errors indicates that they involve precisely the same principles as the branch-point errors considered in the previous portion of the study. Once again, the model selects an incorrect action when noise causes its context representation to resemble a familiar pattern, connected with some other behavioral context, that is associated with that action. The only factor that distinguishes the situation away from branch-points from the branch-point case is that a greater degree of distortion is needed to produce the critical effect. At branch points, very small amounts of context distortion lead to errors, because of the close resemblance between the relevant temporal contexts. At non-branch-point steps, the contexts associated with different actions tend to be more distinct, meaning that a larger distortion of the model's context

representation is needed to produce an error (Figure 26).

To illustrate the point, consider the non-branch-point error described in Table 5 (left). Over the first ten steps shown in the table, the model's output correctly follows the sequence for adding grounds and enters into the sugar-bowl sequence. After picking up the spoon, however, the model moves directly to stirring the coffee, rather than first returning to the sugar bowl, scooping up some sugar, and pouring it into the cup. This error, like errors falling at the beginning of new subtasks, results from a distortion of the context representation that leads it to resemble a pattern usually occurring in a different situation. Here, the latter situation derives from the stirring sequence, where picking up the spoon is followed by fixate-cup and stir. Note that the steps leading up to the fixate-sugarbowl action are quite different from those leading up to fixate-cup in the stirring sequence. This means that the context patterns between which the network must discriminate in this example are rather dissimilar. This, in turn, explains why a higher level of noise is necessary to produce this error than to produce errors at subtask transitions, where the relevant context representations are more similar to one another. Aside from this difference, which is one of degree, the basic factors that lead to error-commission here are identical to those that lead to branch-point errors.

To make the point more broadly, there is a sense in which every step in the sequences the model produces is a "branch

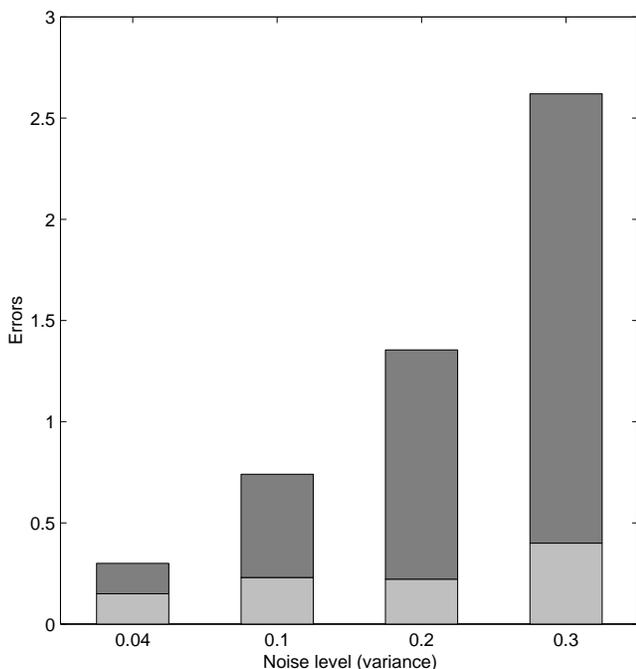


Figure 25. Average number of sequence (light gray) and omission (dark gray) errors per trial at several noise levels. Based on a sample of 100 trials at each noise level.

point.” During training, the model has learned to associate every environmental input with a number of different actions, each associated with a different context. In some cases, two different actions may be associated with very similar contexts. As we have seen, this tends to be the case at the transitions between subtasks. However, even where the relevant contexts are more distinct, the network must use its representation of context to decide among possible actions. The model suggests that the dichotomy between branch points and non-branch points should be replaced by a view according to which each step in a routine falls somewhere on a continuum defined by the distinctiveness of the contexts associated with candidate actions. When the representation of temporal context is disrupted, errors occur first on steps located at one end of this spectrum, where different actions are associated with very similar contexts. With increasing disruption, errors begin to occur at points lying further and further along the continuum.

At high levels of noise another effect can be observed, this one involving not the timing of errors but rather the specific actions the network tends to select. Specifically, with increasing noise, the network shows an increasing bias toward selecting the actions pick-up and put-down and the fixate actions. Pour, tear, stir and the remaining actions are produced with diminishing frequency. This effect is reflected in Figure 27, which shows a step on which the correct response is peel-open. When the context representation is highly distorted, this action becomes rare in comparison with

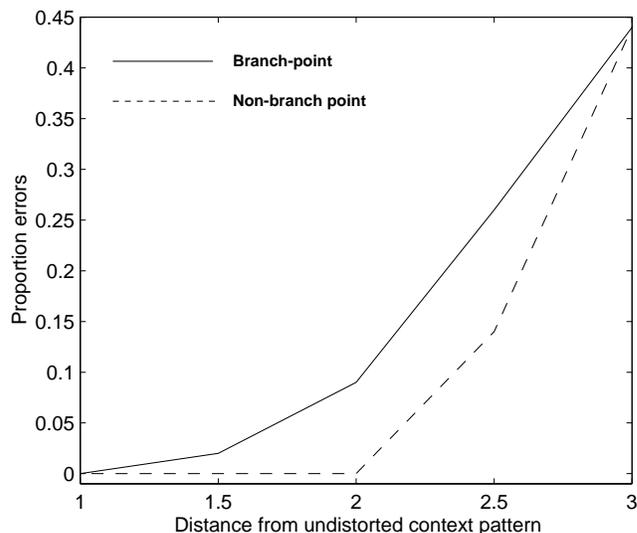
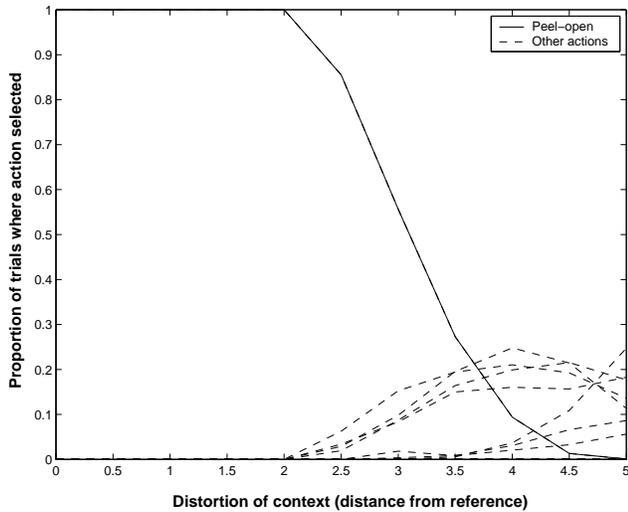


Figure 26. Greater distortion of context is needed to produce an incorrect action at a non-branch-point than a branch-point. The data shown were produced by adding progressively increasing random noise to the context representations normally arising at a branch-point step (the first step of the DRINK subtask in the sequence GROUND → SUGAR (BOWL) → CREAM → DRINK; solid line) and a non-branch-point step (the peel-open step in the sequence GROUND → SUGAR (BOWL) → CREAM; dashed line), and observing the outputs produced by the network. The x-axis shows the Cartesian distance of the distorted context representations from the noise-free pattern (binned in steps of 0.5). The y-axis shows the proportion of trials on which the incorrect action was selected.

put-down, and the fixate actions.

Note that this effect cannot be reduced to a preference for the action most frequently performed in a given environmental context. Indeed, on the step illustrated in Figure 27, the peel-open action itself is the action most frequently associated with the current environmental input during training. Instead, the reason particular actions become predominant derives from the fact that, in the training set, they appear across a particularly wide range of contexts. The pick-up action appears in many contexts because it is a viable option whenever the grasp is empty. The put-down action appears in many because it is legal in all circumstances where the grasp is occupied. The fixate actions are the most generic of all, since there is no situation where they are prohibited. Because the background training set contained examples that include as targets all actions that might plausibly be executed in a given environmental setting, these three actions appeared during training in an extremely wide variety of contexts. Actions more closely tied to specific environmental situations, such as tear, sip or peel-open, appeared in a significantly smaller variety of contexts.

Figure 28 illustrates the relationship between the context-generality of actions and the probability of their being correctly selected. Each point represents a specific action in

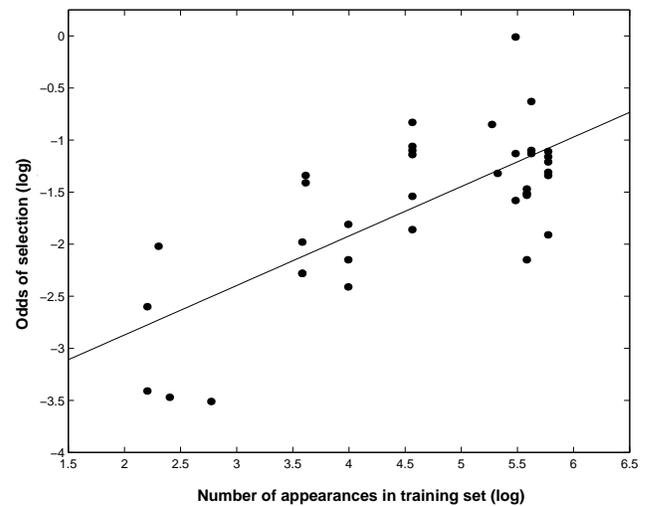


*Figure 27.* Analysis of action selection on the step in the coffee task where the correct action is peel-open, with the carton as the target object. As the distance between the distorted context representation and the reference pattern increases (x-axis), this action becomes decreasingly likely to be selected (solid line). At the same time, the context-general put-down and fixate actions (dashed) become more likely. The x-axis indicates centers of histogram bins. The y-axis indicates the proportion of trials within each bin on which a given action was selected. Actions indicated by dashed lines, in order of increasing activation at the far right of the figure, are fixate-teabag, fixate-coffee, fixate-sugarbowl, fixate-cup, fixate-spoon, put-down, and fixate-sugar.

the sequence GROUND → SUGAR (BOWL) → CREAM → DRINK. On the x-axis is the frequency with which this action occurs among examples used during training (including all versions of the coffee and tea tasks as well as the background examples), as a proxy for context-generality. The y-axis indicates the frequency with which this action was selected on steps in the coffee task when it was the correct action, under noise with very high variance. The network tends to select less frequently actions that appear less often in the training set, and that thus appear in a narrower range of contexts.

To understand this correlation between the context-generality of actions and their robustness, consider what happens when the peel-open step shown in Figure 27 occurs with a highly degraded context representation. If sufficiently distorted, the context will not much resemble the one associated with the peel-open response. Instead, it is more likely to bear slight similarities to a wide range of contexts the network has encountered during training, contexts associated with both this and other external input. If a preponderance of these contexts are associated with a particular output, this output is likely to be selected. The network will therefore tend to produce outputs that are associated with a large range of contexts, i.e. put-down and fixate.

The model’s bias toward context-general actions is one of the factors that leads it to produce a rising proportion of



*Figure 28.* The context-generality effect. Each point corresponds to a step in the sequence GROUND → SUGAR (BOWL) → CREAM → DRINK. The x-axis indexes the number of times the correct action on the relevant step appears as a target output within the training set as a whole. The y-axis indexes the probability that the correct action will be selected on the relevant step. Each data point is based on 1,000 trials, under noise with variance 5 (with unit activations cropped at 0 and 1). Correlation based on log-transformed data yields an  $r^2$  of 0.53. A regression line is shown ( $p < 0.00001$ ).

omission errors with increasing noise. In the coffee-making task, the actions that define whether a subtask has been completed—pouring ingredients into the cup and drinking from it—are all relatively object- and context-specific. As a result of this, these actions are less likely to be selected as noise increases, contributing to the disproportionate increase in omissions. Another, simpler reason for the increasing predominance of omissions has to do with the fact that, with increasing noise, sequential behavior becomes more fragmented. Completion of the individual subtasks in coffee-making—for example, adding cream or sugar—requires that a number of actions be executed in series. Naturally, as performance becomes more and more fragmented, the network develops an increasing tendency to get “side-tracked” before reaching the end of these subsequences, and the goals of the relevant subtasks therefore tend to go unaccomplished.

### Summary of Simulation Results

Together, our simulation results provide evidence in support of the sufficiency of a recurrent network model to account for several basic features of routine sequential behavior. The first part of the simulation demonstrated the ability of the model to maintain a representation of temporal context capable of guiding behavior in circumstances where the environmental situation alone is ambiguous with respect to action, and capable of guiding performance in sequential tasks involving a flexible combination of subtasks. A slight degradation of this internal representation led to errors resembling

everyday slips of action. In keeping with the empirical data, errors affected the ordering of subtasks more strongly than their internal structure. Interestingly, while errors tended to occur at the transitions between subtasks, the processes leading up to such errors were found to begin a number of steps earlier. In fact, noise applied midway through a subtask ultimately proved more disruptive to the model's performance than noise injected at the end of the subtask. This portion of the study also reproduced the reported effect of task frequency on error rate. Further degradation of the model's context representation led to a disruption of sequential structure within subtasks, yielding a pattern of behavior resembling that of ADS patients on a number of levels. Increasing degradation was accompanied by a graded increase in the frequency of independent actions; the errors produced by the model fell into the same categories as those produced by ADS patients; and as in ADS, a correlation was observed between overall error rate and the prevalence of omission errors. High levels of noise had an effect on the particular actions the network tended to select, resulting in a bias toward actions associated with a wide variety of objects and contexts.

### General Discussion

Routine, sequential, object-directed action takes up most of our daily lives. Despite the ubiquity of such action, its computational underpinnings remain incompletely understood. We have presented one account of how routine sequential activities are accomplished, based on the properties of recurrent connectionist networks. In what follows we address the relationship between the present account and traditional models of routine sequence production, discuss some of the model's testable predictions, and consider its implications with respect to a number of key issues in the study of action.

#### *Comparison with Hierarchical Models*

The framework presented here differs from traditional accounts in at least two fundamental ways. First, the structure of the system's sequential behavior emerges from the functional properties of the processing system as a whole, rather than being linked in a direct fashion to discrete elements within the system's architecture. The system produces hierarchically organized behavior without relying on a structurally expressed schema hierarchy. Second, the representations that guide sequential behavior, and the detailed characteristics of the sequencing mechanism itself, develop through experience with relevant task domains rather than being built into the processing system *a priori*. These aspects of the present account allow it to avoid several basic problems associated with traditional models. As discussed in the Introduction, these problems include: (1) difficulty accounting for learning, (2) difficulty specifying a satisfactory sequencing mechanism, and (3) difficulty coping with quasi-hierarchical domains. In what follows, we revisit these three issues, considering them in the light of the present alternative framework.

#### *Learning.*

The account we have presented here indicates how the knowledge structures underlying routine sequential behavior might develop through experience with specific tasks. When faced with the same issue, the traditional framework encounters difficulties. Specifically, it is not clear how individual computational elements within hierarchical models assume responsibility for particular segments of behavior, and how the appropriate links are formed from elements at one level to the levels above and below. The majority of work using hierarchical models circumvents the issue of learning, by building the needed structure into the processing system (e.g., Cooper & Shallice, 2000; Estes, 1972; MacKay, 1985; Rumelhart & Norman, 1982). Such extensive "hand wiring" undermines the explanatory force of hierarchical models (Plaut & McClelland, 2000). In contrast to traditional accounts, the framework we have presented requires only very general assumptions about system architecture, leaving its detailed configuration to the learning process.

One of the few existing proposals for how sequence learning might take place given a hierarchical processing architecture has been put forth by Houghton (1990; see also Grossberg, 1986). The procedure involves activating units representing the individual steps of the target sequence while a compound unit at the next level up in the hierarchy passes through an activation trajectory spanning the full duration of the sequence. During the sequence an associative learning rule is used to strengthen connections between the two levels. In order to implement this learning scheme, Houghton is compelled to assume a mechanism that detects the onset and termination of the sequences to be learned, for example detecting the boundaries of heard words in the context of learning word pronunciations. One problem with this approach is that in many domains there may be no reliable surface markers for event boundaries. As it so happens, Houghton's (1990) chosen domain—speech—provides an excellent illustration of the point. Houghton suggests that word boundaries might be identified based on changes in sound intensity, but in naturalistic connected speech, word boundaries are associated with no such cues (see Morgan & Demuth, 1996). Indeed, there is evidence that the identification of event boundaries, at least in some contexts, depends on some prior knowledge of the sequential structure of the domain (see Avrahami & Kareev, 1994; Zacks & Tversky, 2001). For accounts like the one provided by Houghton (1990), this finding poses a "chicken and egg" conundrum: Acquisition of sequence knowledge depends on the ability to identify event boundaries, but the identification of event boundaries depends on sequence knowledge.

The framework we have proposed faces no such dilemma. To the extent that successful performance depends on information about task segmentation, the model derives this information from the statistical structure of the sequences encountered during learning. No surface indication of event boundaries is required. Indeed, it is significant that recurrent connectionist models have been used to account for the very processes that underlie the identification of event boundaries (Christiansen, Allen, & Seidenberg, 1998; Elman, 1990;

Hanson & Hanson, 1996).

A further difference between the present framework and hierarchical accounts relates not to how learning is accomplished, but rather to what the system learns. In traditional accounts, acquisition of sequence knowledge involves instantiating a locally-represented schema, typically identified with a discrete node or processing unit. In the present account learning results in behavior that reflects schema-like knowledge, but without giving rise to any processing structure corresponding directly to a classical schema. Rather than implementing schema units, the system improves its performance by learning how to preserve and apply task-relevant contextual information. To put it strongly, while the notion of a schema may be useful in describing the system's behavior over time, the system itself contains no schemas at all. In this way, the account we have put forth here mirrors earlier work by Rumelhart et al. (1986, p. 21). As they put it,

In the conventional story, schemata are stored in memory. Indeed, they are the very *content of memory*. In our case, *nothing stored corresponds very closely to a schema*. What is stored is a set of connection strengths which, when activated, have implicitly in them the ability to generate states that correspond to instantiated schemata. This difference is important—especially with regard to learning. There is no point at which it must be decided to create this or that schema. Learning simply proceeds by connection strength adjustment. . . . As the network is reorganized as a function of the structure of its inputs, it may come to respond in a more or less schema-like way.

#### *Sequencing Mechanisms.*

An important aspect of the learning process in the present model is that it leads to sequencing mechanisms that are specially adapted to the tasks the system is to perform. This contrasts with traditional models, which typically assume more domain-general sequencing mechanisms. For example, most hierarchical models rely heavily on reflex inhibition, a mechanism by which units are transiently inhibited following activation. As discussed earlier, this mechanism serves to prevent repeated activation of individual units, allowing activation of the next element in a sequence. Note that reflex inhibition implements an implicit assumption about the sequences the system will need to produce, namely that they will not in general contain repeated elements. However, while this may be true of many naturalistic sequences, it is easy to think of many cases where actions must be repeated: hammering a nail, sweeping the floor, winding a watch, etc. Sequences such as these pose difficulties for systems that rely on reflex inhibition, requiring the addition of supplemental, special-purpose mechanisms. For example, in hierarchical models of spelling, reflex inhibition poses a problem during processing of words with doubled letters, compelling researchers to posit a special “doubling schema,” that serves to override the reflex inhibition mechanism in appropriate cases (Hartley & Houghton, 1996; Rumelhart & Norman, 1982).

Aside from the seemingly *ad hoc* nature of this proposal, it is unclear how it could be applied to sequences such as those just listed, which involve potentially extended series of repeated actions.

The processing framework we have adopted here involves no dilemma between preventing and supporting action repetition. In cases where an action is to be performed only once (e.g., the stir action in our implementation of the coffee task), learning leads the system to inhibit this action once it has been executed. When an action is typically repeated (e.g., the sip action), the system can learn not to inhibit the action after selecting it. Rather than relying on hard-wired, across-the-board assumptions about sequence structure, the system can configure itself in accordance with the detailed sequencing constraints of the particular tasks at hand, constraints that may vary in nature within and between tasks.

As discussed at the beginning of the article, another limitation of the sequencing mechanisms that have been proposed for hierarchical models is that they have difficulty coping with situations where the appropriate action at a given point depends on actions made earlier (recall the example of setting the temperature on a washing machine based on the type of clothing placed inside earlier). Sequencing mechanisms like reflex inhibition preserve very little information about earlier actions, providing an insufficient basis for sequences that involve cross-temporal contingencies. In contrast, the model we have proposed involves a sequencing mechanism that learns to preserve and exploit whatever information about previous events is useful in guiding behavior.

#### *Dealing with Quasi-Hierarchical Structure.*

A major limitation of hierarchical models of sequential action is that they have difficulty coping with situations where details of subtask performance depend on the larger task context, a circumstance very common in human behavior. In the present work, we have shown how a non-hierarchical system can overcome this problem, performing different versions of a subtask in different situations. The model's ability to produce context-sensitive behavior derives largely from its use of distributed internal representations (Hinton, McClelland, & Rumelhart, 1986). Because such representations can capture graded similarity, they are able to encode different versions of a subtask in a way that acknowledges their overlap while still keeping them distinct.

Another way of viewing this aspect of the model is in terms of information sharing among tasks. When a task resembles one that the system has already learned how to perform, the system will “re-use” representations associated with the familiar task in order to perform the new one. Schank and Abelson (1977; see also Schank, 1982) have argued that information sharing of this kind is likely to be involved in the representations underlying human performance. As one example, they discuss the routines involved in eating in restaurants. While different types of restaurant—fancy restaurants, fast-food restaurants, cafeterias—call for different methods for obtaining a table, ordering, paying, etc., there is also a great deal of overlap in the behaviors

they call for. Schank and Abelson argue that this overlap is reflected in the knowledge structures that guide restaurant behavior. Rather than there being a separate “script” for each different kind of restaurant, features common to all restaurants are represented once, with behaviors pertaining to specific kinds of restaurant built on top of this generic set of representations. The framework we have introduced here makes clear how this sort of representational scheme might be implemented, and how it might emerge from experience.

Traditional, schema-based accounts of action have sometimes implemented a form of information sharing through the use of “slots” (e.g., Norman, 1981; Schank & Abelson, 1977). Here, schemas contain variables that can adopt several different specific values. For example, consider the routine of making a peanut butter and jelly sandwich. Here, the actions involved in adding the peanut butter and the jelly are quite similar. One way of understanding performance of the task is to assume a general purpose ingredient-spreading schema, with a slot used to identify the particular ingredient to be spread. While this approach allows a degree of information sharing among related tasks, it entails a sharp distinction between tasks that share information structures and tasks that do not. This can lead to awkward dilemmas when the goal is to address domains where tasks have varying degrees of overlap. For example, while it seems reasonable to posit a single schema for spreading both peanut butter and jelly, it is less clear whether this schema or some other should cover the weakly related tasks of putting icing on a cake, or sauerkraut on a hotdog. In the present framework such dilemmas do not arise, since there is no discrete boundary between tasks that share representations and tasks that do not. The distributed representations the system employs allow it to implement a form of information sharing that is well suited to the fact that tasks may overlap on many different dimensions and to widely varying degrees.

One important feature of this form of information sharing is that it supports generalization. When faced with a novel situation, reasonable inferences can be made about appropriate actions based on the resemblance of the new situation to familiar ones. To return to the restaurant example from Schank and Abelson (1977), someone entering a Wendy’s restaurant for the first time is likely to have a good sense of what to do, based on his or her prior familiarity with other fast-food restaurants. This sort of generalization falls naturally out of the processing framework we have considered here. While our simulations were not designed to illustrate this aspect of the framework, it is frequently evident in the model’s behavior after having made an error. For example, under high levels of noise the model occasionally picked up the sugar bowl and poured sugar directly into the cup. Performing this action placed it in a relatively unfamiliar situation (both in terms of perceptual inputs and temporal context). When this situation arose, however, the model typically responded by seeking out the spoon and performing the stir sequence. This occurs because, while the context following pouring directly from the sugar bowl is unfamiliar, it resembles the familiar situation of having just poured grounds, sugar or cream into the cup, all of which are followed by the

stir sequence.

#### *Accounting for Pathologies of Action.*

Within hierarchical accounts of action, sequencing errors in both normal and apraxic performance have most frequently been understood as reflecting a weakening of the influence of explicit high-level schemas on lower-levels, or of lateral inhibition (Cooper & Shallice, 2000; MacKay, 1985; Schwartz et al., 1991; Shallice, 1988). The present account, in contrast, suggests that action errors result from the degradation of learned, distributed representations of temporal context.

As our simulations demonstrate, this proposal covers many of the same empirical phenomena addressed by the traditional account. However, it also offers some advantages. For example, the present framework appears to provide a more satisfactory account for the *graded* nature of action disorganization. This gradedness is evident in the data concerning recovery in ADS reported by Schwartz et al. (1991; see Figure 7), where the number of independent actions gradually fell over the weeks following initial evaluation. It is also evident across subjects, as in the population of patients reported by Schwartz et al. (1998; see Figure 8). Indeed, a graded continuum of action disorganization appears to connect the slips of normal subjects with the more severe errors of ADS patients (Schwartz et al., 1998). As we have noted, this spectrum appears to begin with disorganization primarily at the between-subtask level, with disorganization progressively infiltrating the within-subtask level as severity increases. It is not clear that this graded progression would fall out of the traditional, hierarchical account. Indeed, in the model of Cooper and Shallice (2000), gradually weakening top-down influence within a hierarchy of schemas resulted in abrupt, non-monotonic shifts in the degree and character of sequential disorganization.

As noted earlier, the Cooper and Shallice (2000) model also failed to capture two other aspects of human errors: (1) the fact that slips of action sometimes involve repetitions of entire subtasks, and (2) the fact that, in ADS, omission errors become increasingly predominant as overall error rate increases. In contrast, the present model reproduced both findings. Furthermore, the model provides a natural account for what Norman (1981) called “capture errors,” lapses from one task into another following a series of actions that the two tasks share. Such errors follow naturally from the fact that in the present model, unlike hierarchical models, each action performed contributes to the system’s representation of temporal context.

An influential idea concerning sequencing errors is that they reflect a failure of “controlled” as opposed to “automatic” processing. For example, Reason (1992) has suggested that naturalistic behavior consists of highly routine subsequences that can be executed without close attention, punctuated by decision points which require the actor to enter a special attentional mode. Norman (1981) makes a similar argument, calling these junctures “attentional checkpoints.” Under this view, errors at the boundaries of subtasks are understood as reflecting a failure to engage the special at-

tentional mechanisms responsible for guiding non-automatic behavior. In contrast to this account, the framework we have put forth involves no sharp distinction between automatic and controlled processing (see also Cohen, Dunbar, & McClelland, 1990), nor any sharp distinction between so-called decision points and other steps in processing. The phenomena that these constructs are meant to address—such as the tendency of slips to occur at the transitions between subtasks—emerge naturally from the system's basic sequencing mechanism.

At the same time that Reason has emphasized the distinction between controlled and automatic processing in explaining slips, he has also employed another idea that is much closer in spirit to the present account. Here, he suggests that errors may often result from "cognitive underspecification," where representations responsible for guiding behavior insufficiently specify the operations to be performed (Reason, 1992). The same theme appears in the work of Norman (1981), which discusses errors due to "insufficient description" (a term adopted from Norman & Bobrow, 1979). In essence, these accounts suggest that slips may occur due to internal representations that are in some way vague. The present account cashes out this intuition, making computationally explicit what this representational imprecision might involve. "Underspecification" emerges in the present account when the system's distributed representation of context is disrupted, producing a pattern that bears partial resemblances to familiar patterns linked to different behavioral contexts. In his arguments concerning cognitive underspecification, Reason (1992) has emphasized that "when cognitive operations are underspecified, they tend to default to contextually appropriate, high-frequency responses" (p. 71). The modeling work we have presented—in particular, the simulation addressing the relationship between task frequency and lapse errors—provides a mechanism-based explanation for why this is so.

### *Comparison with Production System Models*

While we have focused on hierarchical models, it is important to note that there is a second tradition in the modeling of sequential behavior, the production system approach. Unlike the models we have already discussed, production systems such as ACT-R (Anderson & Lebiere, 1998) and Soar (Laird, Newell, & Rosenbloom, 1987; Newell, 1990) do not typically adopt architectural hierarchy as a core assumption. Nevertheless, a strictly hierarchical processing scheme is often built into such systems. This is particularly evident in the way that production systems typically implement goal representations. As preconditions for the firing of productions (e.g., Anderson & Lebiere, 1998), goal representations often play a role virtually identical to the goal-nodes in the Cooper and Shallice (2000) model, mediating between discrete, independent levels of task organization. In many paradigms, including both ACT-R and Soar, this role is made explicit through the use of a goal stack, onto which high-level goals are "pushed" first, followed by goals at successively lower levels.

Because of this implicitly hierarchical processing scheme, production systems face many of the same limitations we have attributed to explicitly hierarchical models. First, there are issues related to learning. Like other hierarchical models, production systems contain discrete processing elements that relate to extended segments of behavior (e.g., in ACT-R, productions for setting task-level goals). Once again, this makes it necessary to posit special mechanisms to explain how and when new processing elements are created and configured (e.g., ACT-R's theory of production compilation, Anderson & Lebiere, 1998). Also like hierarchical models, production systems face problems stemming from their reliance on mechanisms that build in *a priori* assumptions about task structure. The most serious concerns relate to the goal stack construct mentioned above. While this mechanism allows production systems to address tasks with multiple levels of structure, Altman and Trafton (1999) have argued that it faces difficulty in addressing some aspects of how human subjects actually accomplish such tasks (see, e.g., Van Lehn, Ball, & Kowalski, 1989) and some of the errors that such subjects commit (e.g., Byrne & Bovair, 1997).

Quasi-hierarchical task structure also raises problems for production system models. In general, production systems use slot-filling to implement information sharing among representations subserving interrelated tasks. As discussed above, this approach requires that categorical decisions be made about the relatedness of activities, a requirement that becomes awkward in naturalistic domains where tasks may interrelate in a graded fashion and at a variety of levels.

Our own theory places a strong emphasis on the potential for representations to partially overlap, arguing that this plays a central role in both enabling correct performance and predisposing the system to errors. It should be noted that representational overlap also plays a key role in the ACT-R framework. Here, productions send activation to representations stored in memory to the degree that the features of such representations match a search template; excitation is received by representations that match the template only partially (Anderson & Lebiere, 1998). Interestingly, this aspect of the theory was first introduced in the context of efforts to address action slips (Lebiere, Anderson, & Reder, 1994). Here, as in the theories of Reason and Norman, errors result when an inappropriate operation is selected because its calling conditions are partially met, a proposal that resonates with the account we have presented in the current work.

### *Predictions*

The modeling work we have presented gives rise to a number of testable predictions. In the following sections, we discuss predictions relating to normal performance, slips of action, and ADS.

#### *Predictions Concerning Normal Performance.*

One distinctive aspect of our model is that it is sensitive to similarities among different temporal contexts. This sensitivity is most clearly evident in the model's errors, which can often be understood as involving a confusion between the present temporal context and some other familiar context

to which it bears a resemblance. However, this property of the model is also manifest even when it does not commit errors, in biases that emerge in its sequencing behavior. For example, in working with the model we noted that, following training on the coffee-making task alone, it showed a tendency to add cream before sugar. This was surprising at first, because fully half of the sequences presented during training involved sugar being added first. The bias turns out to derive from the resemblance between two subtask sequences: the sequence for adding grounds and the one for adding sugar from the sugar packet. These two subtasks share many of the same actions, and the objects involved are also similar (in particular, the sugar packet shares features with the coffee packet). This resemblance causes a corresponding similarity between the context representations arising immediately upon completion of the two sequences. With this in mind recall that, during training, the model learns to transition from adding the grounds into either adding sugar or adding cream. At the same time, it learns to transition from adding sugar to adding cream. Because the grounds sequence resembles the sugar-packet sequence, and the latter is associated with a transition to cream-adding but not sugar-adding, the model develops a bias toward adding cream after adding grounds.<sup>7</sup>

The model predicts that such effects should be apparent in the behavior of human subjects learning to produce appropriately structured sequences in the laboratory. Specifically, it should be possible to observe biases in the sequences produced by such subjects, based on an interaction between the objective structure of the target sequences and resemblances among the subsequences involved.

#### *Predictions Concerning Slips of Action.*

While the model reproduced the finding that slips of action tend to occur at the transitions between subtasks, surprisingly such errors were more likely to occur if the model's representation of context was perturbed near the middle of a subsequence than at its end. Thus, the model predicts that human subjects should show the same pattern of sensitivity; distraction toward the middle of a subtask sequence should give rise to more frequent errors at the transition to the next subtask than distraction closer to that transition point.

A second prediction has to do with the tendency in the model for increasing noise to erode temporal structure beginning at the global level and extending to the internal structure of subsequences only later. The same pattern should be observed in normal subjects performing componentially-structured sequential tasks under conditions of increasing distraction.

#### *Predictions Concerning ADS.*

Our simulation results give rise to the prediction that ADS patients should show a tendency toward context-general actions, actions that are associated with a large range of task contexts and environmental configurations. As in the model, such actions are likely to include ones that respond to objects' basic mechanical affordances, including simply picking objects up and putting them down. Actions that relate to objects' conventional uses, or that require specific config-

urations of the environment are predicted to occur relatively infrequently. The model's predictions in this regard appear to receive some advance support from the informal observation of Schwartz et al. (1991) that ADS patients spend a great deal of time simply "toying" with objects. However, a more formal test of the prediction remains to be conducted. A related prediction is that individuals with ADS should engage in frequent and prolonged periods of visual scanning during the performance of sequential tasks, since such action is itself highly context-general. Informal observations are consistent with this prediction (M. Schwartz and L. Buxbaum, personal communication).

It should also be possible to observe a bias toward context-general actions in normal subjects under conditions of distraction. Data that may be related to this prediction have been recently reported by Creem and Proffitt (2001). Here, subjects were asked to pick up individually presented familiar objects, each of which had a handle of one kind or another. In an initial test, Creem and Profit found that subjects lifted these objects using their handles, even when the handle was oriented away from them. However, under conditions of distraction (concurrent performance of a paired associate task), subjects tended to pick up objects without using their handles, grasping them elsewhere instead. While the task used in this experiment is not sequential in the usual sense, the results reported accord with the predictions of the present model by providing evidence that distraction can influence the affordances to which subjects respond.

#### *Modeling Naturalistic Action: General Considerations*

Our primary focus in the present work has been on sequencing phenomena. However, the framework we have presented also speaks to a number of other central issues in the domain of routine sequential action. One set of issues relates to the fact that much of everyday action is carried out upon objects. This raises the questions of how objects are selected as targets of action, and how the perception of objects may in turn impact the selection of actions. Another set of issues relates to the practical, goal-oriented nature of most routine sequential activity. This raises the question of how goals are represented, and how the relevant representations structure behavior. In what follows, we consider how these issues are dealt within the present computational framework.

#### *Selection for Action.*

The present model adopts the view that object selection and action selection involve very similar procedures: Objects are first selected by performing perceptual actions, and then acted upon by executing manipulative actions, resulting in

<sup>7</sup> As indicated by data presented earlier, training the network on both the coffee and tea tasks produced the opposite sequencing bias. Here, the network tended to complete the sugar sequence before the cream sequence. In this case, the bias represents a frequency effect. Across the four versions of the coffee task and the two versions of the tea task, the stirring sequence is more frequently followed by the sugar subtask than the cream subtask.

what Hayhoe (2000) has referred to as a “perception-action sequence.” While this provides a framework for understanding a number of empirical findings relating to action with objects (as discussed in the Introduction), it does not answer the question of how the system determines which specific object to select at any given point in performance. According to the present account, this is accomplished by learned associations between particular environmental inputs and internal contexts on the one hand, and specific perceptual actions on the other. Note that here, once again, the mechanisms underlying object selection are the same as those underlying the selection of manipulative actions. An interesting consequence of this is that disruptions of context representation affect not only the selection of overt actions, but also the selection of target objects. This provides a potential explanation for the fact that, in both everyday slips of action and in apraxia, sequencing errors tend to occur alongside errors in object selection. Indeed, in our simulations the majority of subtask omission and repetition errors began with errors in object selection (inappropriate perceptual actions).

The framework we have introduced leaves room for several more detailed aspects of object selection not implemented in the specific model reported here. For example, in the present version of the model there is a one-to-one correspondence between perceptual actions and target objects (e.g., fixate-carton). A fuller model would specify sought-for items using a feature-based representation, in line with current accounts of visual search (Duncan, 1996; Mozer & Sitton, 1996; Mueller, Humphreys, & Donnelly, 1994; Treisman, 1988). Pursuing this would raise important questions about what specific feature dimensions might be relevant to search in naturalistic tasks. While basic perceptual features such as shape, color, and location are clearly used in indexing objects for search, it is possible that action-related aspects of objects might also serve as a basis for locating them. For example, it is possible that attention may be drawn to objects that afford an action currently planned (Humphreys & Riddoch, 2001; Rizzolatti, Craighero, & Sieroff, 1998). More abstract associations between objects and whole lines of goal-directed behavior could also be important; according to Duncan (1993), “though it is conceivable that simple stimulus features (colour, motion, etc.) could often be a clue to ‘importance’, it must still be true that [stimulus] selection is in a sense controlled by ‘meaning’, involving at least implicit reference to the consequences of several different lines of behavior” (p. 63).

In the model presented here, most perceptual actions have deterministic results. If fixate-carton is selected, fixation moves to the carton with perfect reliability. However, in actual search one may encounter non-target objects before attaining one’s target. For example, in looking through the refrigerator for a carton of cream, one may first notice a milk carton or an orange-juice carton, in addition to other less closely related objects, before finally discovering the cream. Adding this aspect of search to the model would involve addressing two questions. First, which non-targets are most likely to be attended during visual search? Presumably, one is more likely to select objects that share features with the

target object, but what features are relevant? Second, how does the system decide when it has found the object sought for, rather than some distractor? We assume that such verification could be accomplished based on the model’s internal representation of context, which carries information about the preceding perceptual actions. Exploring the implications of this proposal will provide an interesting goal for future work.

The fact that we have not implemented feature-based or probabilistic search in the present model may partially account for the relative rarity in our simulations of object substitution errors (beyond those occurring at the beginning of branch-point errors). As noted earlier, such errors were somewhat less frequent than in the behavior of normals and apraxics. We assume that some errors in object selection derive from the unreliable nature of search and its sensitivity to similarities among objects, and that building these aspects of search into the model would result in more frequent and glaring errors in object-selection. This being said, there are other aspects of the present simulation that may also account for the scarcity of object substitutions. Most important is the small number of objects included in the modeled environment, and the restricted range of activities on which the model was most heavily trained. A larger environment and a larger behavioral repertoire are likely to have resulted in more frequent object substitutions, even using the present simplified implementation of object selection.

#### *Responding to Objects.*

Another key role for objects in the context of sequential routines is as *triggers* of action. A wide range of evidence indicates that action selection is directly influenced by perceived objects. The performance of subjects in laboratory tasks such as the Stroop (MacCleod, 1991) and Eriksen flanker (Eriksen & Eriksen, 1974) tasks supports the idea that the perception of a visual stimulus leads to activation of strongly associated responses (e.g., viewing a word may activate the verbal response associated with reading it aloud). Other data indicate that this phenomenon extends to complex stimuli familiar from everyday life. For example, Tucker and Ellis (1998) had subjects view photographs of familiar objects with handles, such as pots and pans, asking them to indicate with a left- or right-handed button press whether the object was displayed right-side-up or upside-down. Results showed that subjects were fastest to respond when the handle of the object appeared on the same side of space as the responding hand, as if viewing the object led to activation of the reaching movement that the object would ordinarily invite (see also Riddoch et al., in press, 1998).

The ability of objects to act as triggers of action is integral to the model we have proposed. During training, the model learns to associate perceptual inputs with particular actions. Presenting a given object to the fully trained model leads directly to activation of the associated responses. Of course, in the model as in human behavior, any given object or scene is likely to be associated with multiple responses. Action selection is thus a matter of selecting among the actions afforded by a given perceptual input. In the model, this is accom-

plished by combining information about the current stimulus with internally-maintained context information. The latter acts in effect as a filter on the external input, allowing it to trigger only the action appropriate to the present context.

This aspect of the model's function links it closely to several important models of cognitive control, which cast control as a top-down input biasing the system toward context-appropriate responses to incoming stimuli (e.g., Cohen et al., 1990; Norman & Shallice, 1986). An interesting aspect of such models, also shared by the present one, is that the context or control signal can specify a *class* of stimulus-response mappings, while allowing the system to select a specific response on the basis of perceptual inputs. Thus, in the model of the Stroop task proposed by Cohen et al. (1990), the system's representation of context can bias it toward color-naming or word-reading responses, but in either case selection of a specific output is driven by the stimulus word presented. In effect, the internal representation of context provides a broad specification of the actions to be performed, allowing the specifics to be worked out based on the detailed characteristics of the environmental input. Such an arrangement seems likely to be involved in routine actions on objects, where each execution of a given type of action must be fine-tuned to highly contingent aspects of the environment. For example, turning on the lights upon entering a room may involve flicking a switch, turning a dial, pushing a slider, pressing a button, etc., and in each case the specifics of movement must be adapted to the detailed metrical properties of the available device. The framework we have presented here points to an account of how the action system might deal with such situations, using an internal representation of context to specify the broad class of action to be performed while allowing environmental inputs to determine the details of performance.

To this point, we have focused on the role of objects in triggering individual actions. However, in our model, object perception can also trigger entire behavioral sequences. In fact, this occurs on each test run of the model; the perceptual input (fixated: cup, 1-handle, clear-liquid; held: nothing) is sufficient to evoke an entire coffee- or tea-making sequence. This ability of external inputs to trigger extended behavioral sequences fits well with human behavior. Duncan (1996) has emphasized that, because the environment is not entirely predictable, adaptive behavior requires the ability to enter new lines of behavior in reaction to environmental contingencies. An exaggeration of this normal tendency is seen in so-called *utilization behavior* (Duncan, 1986; Lhermitte, 1983; Shallice, Burgess, Schon, & Baxter, 1989). In this syndrome, typically observed in the setting of frontal lobe pathology, patients tend to undertake activities suggested by objects in their environments, even when such activities are inappropriate to the broader context. For example, Lhermitte (1983) described a patient who, upon observing a painting, a hammer and some nails lying on the floor of the testing room, proceeded to use the tools to hang the picture. The framework presented here suggests that such behavior might be understood, like related forms of action pathology, as stemming from a degradation of context information, resulting in

a bias toward high frequency "default" behaviors.

A final set of points concerning the role of objects in the present framework has to do with the way objects are represented by the processing system. The model's internal representations develop in order to facilitate the mapping from perceived objects to actions. One consequence of this is that objects that afford a similar set of actions become associated with similar internal representations.<sup>8</sup> This fits well with experimental data showing that human subjects tend to categorize objects on the basis of similarities among the actions they invite (Rosch, Mervis, Gray, Johnson, & Boyes-Braem, 1976). Another consequence of the learning process is that the system's internal representation of objects is more strongly influenced by perceptual features that predict afforded actions than by other features. This gives the processing system the potential to infer uses for objects, based on resemblances to other objects. This capacity is frequently displayed in human behavior, where objects are often applied to novel ends. For example, lacking the most appropriate tools, one might stir a drink using a knife or pound a nail using a wrench. People are also able to make reasonable inferences about the actions afforded by an unfamiliar object (e.g., guessing that one might pluck or strum a zither, even if one has never seen one before).

The ability of human subjects to distinguish action-relevant from action-irrelevant object features is highlighted in experiments by Brown (1990) and Chen and Siegler (2000). Here, toddlers first learned how to retrieve a toy using a novel tool. They were then faced with the same task again, but were required to select from a new set of tools, some of which resembled the original tool along different perceptual dimensions. Successful generalization required the toddler to differentiate between features relevant to performing the task (e.g., tool length and shape) and unimportant features (e.g., surface decoration). The model we have presented provides a basic account of how this differentiation might be made. Again, since the model's internal representations develop in order to support action selection, they tend to amplify information about each stimulus that is most informative for determining which actions it affords, and to deemphasize stimulus features that are irrelevant. Thus, when presented with a novel object, the model will tend to infer its uses based on feature dimensions that have been informative in prior learning (see Rogers & McClelland, submitted, for similar ideas applied to the domain of semantic memory).

It is significant that the model's learning about objects takes place entirely within the context of extended tasks. As a result, its internal representations integrate information about perceived objects with information about temporal context. One interesting implication of this is that degrading context information can impair the model's ability to represent the actions afforded by objects. In our simulations of ADS, for

<sup>8</sup> Although the point can be demonstrated using the model presented here, earlier work already clearly shows how internal representations in connectionist networks are influenced by the similarity structure of output representations (see McClelland et al., 1995; Rogers & McClelland, submitted).

example, severe degradation to the model's representation of context biased against selection of context-specific actions, even when these were the actions most frequently associated with the presently fixated object. Context representation is equally dependent on object representation; impairing the model's ability to encode the features of objects should challenge its ability to maintain a useful representation of temporal context.

Bearing in mind this tight interdependence between object and context representation, it is interesting to consider an ongoing debate in the literature on ideational apraxia, a syndrome closely related to ADS. Some researchers have proposed that this form of apraxia primarily involves impaired knowledge concerning the use of objects (e.g., DeRenzi & Lucchelli, 1988). In contrast, others have suggested that the syndrome derives from impaired knowledge concerning the sequential structure of temporally-extended tasks (e.g., Lehmkuhl & Poeck, 1981). While our model does not directly address the data that inform this debate, it suggests that knowledge concerning object use and the knowledge that supports sequencing may be intertwined. Degradation to representations in either domain may manifest as impaired performance in the other, and damage at some levels of the processing system may impact both.

#### *Representing Goals.*

It is frequently observed of human sequential behavior that it is organized around goals (e.g., Cooper & Shallice, 2000; Duncan, 1993; Fuster, 1989; Miller et al., 1960). Many observable aspects of behavior support this view: People often treat as interchangeable strategies that yield the same results; they monitor the outcome of their activities, evaluating whether they have brought about intended effects; and they compensate for unanticipated obstacles to action in a way that seems oriented toward the accomplishment of specific ends. In response to such behavior, some models of action incorporate special mechanisms for representing goals. Miller et al. (1960) posited "TOTE" (test, operate, test, exit) units as basic processing elements, one function of which is to compare the state of the environment with a goal state. Cooper and Shallice (2000) employed "goal nodes" as gates on activation flow between schemas at adjacent hierarchical levels (see Figure 3). Explicit goal representation also plays a central role in production system models of action (Anderson & Lebiere, 1998; Laird et al., 1987; Newell, 1990).

While these efforts to address the goal-directedness of action highlight an important set of psychological and computational issues, they share at least two limitations. First, most existing computational accounts minimize the extent to which goals may be context dependent (a point emphasized by Agre, 1988). For example, one's goals in cleaning the house may vary widely depending on whether one is just tidying up or preparing for a visit from one's mother. Most existing models that depend on goal representations make no allowance for this context dependence. Second, there are many types of routine behavior for which it is not straightforward to identify discrete, explicit goals, for example taking a walk, playing the violin, or attending church.

In contrast to traditional models, the model we have presented here does not rely on special goal representations to structure its behavior. The model enters into structured lines of activity not in response to the explicit setting of a goal, but because previous events have placed it in an internal state and environmental context that predisposes it toward those activities. In our simulations, for example, entering into the cream-adding sequence does not involve explicitly setting the goal of having cream in the coffee. Instead the model simply enters this sequence when it detects the appropriate context, based on earlier learning. Because the framework we have introduced does not depend on explicit goals, there is no problem in applying it to behaviors with non-obvious goals. Thus, understanding the activities of someone going outside for a walk does not require identification of the goals that prompted them to enter the relevant activities. On the present account, walk-taking could be triggered by feelings of restlessness, a desire to breathe fresh air, the arrival of two o'clock (if one is in the habit of taking a walk then), or any other appropriate context.

However, while our model involves no special mechanisms for representing goals, it can produce behavior that appears goal-directed. For example, the model can learn to persist in a given action until a particular environmental criterion is met. This is apparent during the drinking sequence in the coffee-making task. During training, the model learns to take two consecutive sips, after which the cup is always represented as empty, and the correct next action is *say-done*. Following training, if the cup is presented as still containing coffee after the second sip, the network will produce another sip action, and it will continue to repeat this action until the cup is represented as empty. The model here implements something like the TOTE cycle described by Miller et al. (1960), continuing an activity until a particular goal is achieved.

Similarly, the model can learn to treat as interchangeable action sequences that address the same goal. For example, in our simulations, the model learned that the sugar-packet and sugar-bowl sequences could be used interchangeably. Thus, the model learned to associate the same contexts with both versions of the sugar subtask. While, in this case, the model was directly trained to perform two interchangeable sequences in the same set of contexts, systems of this sort can also infer sequence equivalence, interchanging equivalent sequences in a way that produces overall sequences the network has not observed during training (unpublished observations).

While the present model implements the view that organized action can occur without explicit goals, it seems clear that in some circumstances human action does involve instantiation of explicit goal representations. In principle, the present model could be elaborated to address such situations, possibly by including connections between the model's hidden units and a new group of units dedicated to representing desired states of the system or environment. Thus, the account we have presented is not intended to deny the existence or psychological importance of explicit goal representations. Nonetheless, it does treat skeptically the idea that such repre-

sentations are fundamental to all routine sequential behavior (see also Agre, 1988).

### *Implications for Related Domains*

Among the many questions raised by the work we have presented, an important one concerns the extent to which the present model is relevant to non-routine sequential behavior, for example that involved in problem-solving, planning, error-detection and compensation, and coordination of multiple tasks. It is likely that in order to account for these aspects of behavior, additions to the model would be necessary. For example, the demands of planning appear to require some means of forecasting the outcome of one's actions, suggesting the addition of a forward model to the present architecture (Jordan & Rumelhart, 1992; see also Elsner & Hommel, 2001). It is possible that other elements, for example a mechanism supporting episodic memory, may also be necessary to support non-routine behavior (Cohen & O'Reilly, 1996; Ericsson & Kintsch, 1995). Nonetheless, we speculate that much of cognition and behavior, even in such domains as reasoning and problem-solving, may share a basic reliance on mechanisms of the sort illustrated in present model. It has been proposed that the cognitive operations involved in problem solving may themselves take the form of familiar, if very general-purpose, routines (see, e.g., Anderson & Lebiere, 1998; Chapman & Agre, 1987). Thus, at some level, problem solving itself may be a form of "routine" behavior. To the extent that this is the case, the mechanisms discussed in the current work may also be relevant to understanding the cognitive operations involved in problem solving and other non-routine behavior.

A related issue concerns the relevance of the current work to dysexecutive syndromes (Duncan, 1986; Milner & Petrides, 1984), a set of clinical conditions involving a specific disruption of relatively non-routine tasks involving planning, problem solving, and preservation and manipulation of information. While the present modeling work was not explicitly addressed to this domain, it nonetheless yields an account according to which damage to the processing system most strongly affects tasks that are less frequently performed or that rely strongly on the accurate preservation of contextual information. This suggests the possibility that the present account may be relevant to understanding executive disorders as well as disorders of everyday action. In this connection, it is interesting to note that prefrontal cortical damage, the lesion most closely associated with executive disorders, has also been shown to predispose to capture errors analogous to those observed in everyday slips of action (Della Malva, Stuss, D'Alton, & Willmer, 1993), and to impair performance in tasks requiring the ordering of actions based on "script" knowledge (Godbout & Doyon, 1995; Grafman, 1995). Indeed, so-called "frontal apraxia" (Luria, 1966) can closely resemble action disorganization syndrome, and some reported cases of action disorganization syndrome have involved prominent frontal lesions (e.g., Schwartz et al., 1991).

These comments raise the question of how our model re-

lates to what is known about the functional neuroanatomy underlying sequential action. Neuropsychological, neurophysiological and neuroimaging data point to a central role for several brain areas, including portions of frontal cortex (e.g., Fuster, 1995; Grafman, 1995, and see above), parietal cortex (e.g., DeRenzi & Lucchelli, 1988), the cerebellum and basal ganglia (see Hikosaka et al., 1999, for review). While the account we have presented does not attempt to delineate the division of labor among these brain areas, it does specify a set of computations that they may collaboratively support. Specifically, it suggests that this network of brain areas works together to maintain a representation of temporal context, integrating this with perceptual inputs in order to facilitate response selection. In this regard, it is interesting to note that features of the model we have presented bear a close resemblance to features of existing computational models addressing the roles of prefrontal cortex (Cohen & Servan-Schreiber, 1992) and basal ganglia (Berns & Sejnowski, 1998).

Another set of issues relates to knowledge acquisition. The model we have presented acquired its behavior through learning, and it is central to the intended account that the configuration of the model is shaped by the structure of the tasks it is asked to perform. Nonetheless, the present model is not intended as a literal account of the learning process as it occurs in naturalistic human behavior. Thus, another challenge for future work will be to adapt the training corpus, learning algorithm, and processing architecture to more closely reflect the actual events involved in the acquisition of sequential behaviors.

A final set of questions involves the relation between the behavioral phenomena addressed in the current work and other forms of routine sequential behavior. Models very similar to the one presented here have been proposed in work on language comprehension (Elman, 1991; McClelland et al., 1989) and production (including errors; Dell et al., 1993, 1999), raising the intriguing possibility that sequence production in language may rely on the same mechanisms as non-linguistic sequencing. In agreement with some others (e.g., Gupta & Dell, 1999), we suspect that a common set of computational principles and mechanisms underlie both linguistic and non-linguistic behavior. However, as will be clear from the foregoing discussion, the mechanisms in question adapt their behavior to the detailed structure of particular action domains. The principles captured in the recurrent connectionist framework thus have the potential to play out in very different ways in the realms of language and non-linguistic action.

## Conclusion

The domain of routine sequential activity raises a rich set of psychological issues, engaging the areas of perception, motor control, attention, and memory. A particularly central and poorly understood issue concerns how the cognitive system constructs and utilizes a representation of time-varying task context. The roughly hierarchical structure of many everyday tasks has led numerous theorists to the idea that the processing system itself assumes a hierarchical structure.

We have proposed an alternative framework for understanding routine sequential action, which overcomes several basic limitations of the hierarchical approach. Here task structure is represented not at the level of system architecture, but rather in the distributed representations the system uses in performing particular tasks. The system arrives at these internal representations through learning, leading to the emergence of sequencing mechanisms that are flexible, context-sensitive, and responsive to graded similarities among tasks.

Needless to say, the account we have presented abstracts over a great deal of important detail. Where possible, we have pointed to directions in which the model could be further developed in order to better capture detailed aspects of human behavior. Another limitation of the modeling efforts described here derives from the empirical data they address. Data concerning routine sequential behavior is scarce, and much of the available information is qualitative or anecdotal. In view of this, we consider it an important aspect of the present model that it makes several detailed and testable predictions concerning human sequential behavior.

Finally, one welcome aspect of existing research on routine sequential action is the degree to which relevant theories have been proposed in the form of explicit, implemented computational models. It is our hope that the work we have presented here will encourage the continuation of this trend.

## Appendix: Coding of Model Performance

### *Counting of Independent Actions*

In order to quantify the frequency of independent actions produced by the model, we adapted the coding scheme devised by Schwartz et al. (1991). The latter involves two steps: (1) coding of individual actions, and (2) a “bracketing” procedure according to which actions are grouped into subtask units centering on the achievement of a goal. Independent actions are ones that fall outside bracketed groupings.

#### *Listing of Actions.*

Following the approach used by Schwartz et al. (1991), for each trial an initial listing was made of the sequence of actions (designated as “A1” in the Schwartz et al. work). Only overt actions were listed; fixate actions were omitted. In keeping with the details of the scheme of Schwartz and colleagues, pick-up actions were explicitly listed only if two or more cycles of processing elapsed prior to any action being performed with the object involved, and put-down actions were listed only if they occurred two or more cycles after the last action performed with the object. A single pick-up, put-down action was coded if an object was picked up and then put down within two time steps, with no action being performed using the object.

As in Schwartz et al. (1991), actions in the resulting listings were classified according to the subtask to which they related: GROUND, SUGAR (BOWL), SUGAR (PACK), CREAM, STIR, or DRINK. A residual category for unassignable actions was included but rarely used.

#### *Bracketing.*

Schwartz and colleagues used this term for a procedure for grouping A1's that lead up to the achievement of a “crux” action, an action that represents the completion of a given subtask. Independents are A1s that lie outside any bracketed group. The bracketing procedure we followed was based closely on the one described in Schwartz et al. (1991). Actions were grouped together if they derived from the same subtask and led up to a crux action (sipping, stirring, or pouring of coffee-grounds, cream, or sugar). Groups of actions were not grouped together if separated by a crux action—or more than one consecutive non-crux action—from another subtask. Following the bracketing procedure, each action falling outside of a bracketed group was counted as an independent.

### *Coding Scheme for Sequence and Omission Errors*

To some extent, our categorization of the errors made by the model was made informally. However, a more explicit coding scheme was devised for the purposes of quantifying omission and sequence errors. Our approach was based on that adopted by Schwartz and colleagues in their empirical studies of ADS patients (e.g., Schwartz et al., 1991). This involved establishing a list of specific errors, based on a combination of preliminary observations of the network's performance and a consideration of the logical possibilities presented by the normative sequences. The lists used in generating the data presented in the paper were as follows:

#### Sequence errors

- Drinking prior to addition of all ingredients  
(if followed by additional ingredients)
- Repeated addition of any ingredient  
(with one or more intervening actions)
- Pouring cream prior to opening carton
- Scooping with spoon without opening sugar bowl
- Adding coffee only after adding other ingredients
- Stirring prior to adding any ingredients
- Repetition of stirring action  
(more than two consecutive stir actions)

#### Omission errors

- No coffee grounds added
- No sugar added
- No cream added
- Drinking omitted
- Ingredient added but not stirred in

## References

- Agre, P., & Chapman, D. (1987). Pengi: An implementation of a theory of activity. In (p. 196-201).
- Agre, P. E. (1988). *The dynamic structure of everyday life* (Tech. Rep. No. 1085). Artificial Intelligence Laboratory, Massachusetts Institute of Technology.
- Allport, A. A. (1987). Selection for action: Some behavioral and neurophysiological considerations of attention and action. In H. Heuer & S. A. F. (Eds.), *Perspectives on perception and action* (p. 395-419). Hillsdale, NJ: Lawrence Erlbaum Associates.
- Altman, E. M., & Trafton, J. G. (1999). Memory for goals: An architectural perspective. In (p. 19-24).
- Anderson, J. R., & Lebiere, C. (1998). *The atomic components of thought*. Hillsdale, NJ: Erlbaum.
- Avrahami, J., & Kareev, Y. (1994). The emergence of events. *Cognition*, 53, 239-261.
- Baars, B. J. (1992). The many uses of error: 12 steps to a unified framework. In B. J. Baars (Ed.), *Experimental slips and human error: Exploring the architecture of volition* (p. 3-34). New York: Plenum Press.
- Ballard, D. H., Hayhoe, M. M., Li, F., & Whitehead, S. D. (1992). Hand-eye coordination during sequential tasks. *Philosophical Transactions of the Royal Society, London (B)*, 337, 331-339.
- Ballard, D. H., Hayhoe, M. M., Pook, P. K., & Rao, R. P. N. (1997). Deictic codes for the embodiment of cognition. *Behavioral and Brain Sciences*, 20(4), 723-767.
- Berns, G. S., & Sejnowski, T. J. (1998). A computational model of how the basal ganglia produce sequences. *Journal of Cognitive Neuroscience*, 10(1), 108-121.
- Brown, A. L. (1990). Domain-specific principles affect learning and transfer in children. *Cognitive Science*, 14, 107-133.
- Brown, G. D. A., Preece, T., & Hulme, C. (2000). Oscillator-based memory for serial order. *Psychological Review*, 107(1), 127-181.
- Buxbaum, L. J., Schwartz, M. F., & Montgomery, M. W. (1998). Ideational apraxia and naturalistic action. *Cognitive Neuropsychology*, 15, 617-643.
- Byrne, M. D., & Bovair, S. (1997). A working memory model of a common procedural error. *Cognitive Science*, 21, 31-61.
- Chapman, D., & Agre, P. E. (1987). Abstract reasoning as emergent from concrete activity. In M. P. Georgeff & A. L. Lansky (Eds.), *Reasoning about actions and plans: Proceedings of the 1986 workshop* (p. 411-424). Los Altos, California: Morgan Kaufmann Publishers.
- Chen, Z., & Siegler, R. S. (2000). Across the great divide: Bridging the gap between understanding of toddlers' and older childrens' thinking. *Monographs of the Society for Research in Child Development*, 65(2).
- Christiansen, M. H., Allen, J., & Seidenberg, M. S. (1998). Learning to segment speech using multiple cues: A connectionist model. *Language and Cognitive Processes*, 13, 221-268.
- Christiansen, M. H., & Chater, N. (1999). Toward a connectionist model of recursion in human linguistic performance. *Cognitive Science*, 23(2), 157-205.
- Cleeremans, A. (1993). *Mechanisms of implicit learning*. Cambridge, Massachusetts: MIT Press.
- Cohen, J. D., Dunbar, K., & McClelland, J. D. (1990). On the control of automatic processes: A parallel distributed processing account of the stroop effect. *Psychological Review*, 97(3), 332-361.
- Cohen, J. D., & O'Reilly, R. C. (1996). A preliminary theory of the interactions between prefrontal cortex and hippocampus that contribute to planning and prospective memory. In M. Bradimonte, G. O. Einstein, & M. A. McDaniel (Eds.), *Prospective memory: Theory and applications* (p. 267-295). Mahwah, New Jersey: Lawrence Erlbaum Associates.
- Cohen, J. D., & Servan-Schreiber, D. (1992). Context, cortex, and dopamine: A connectionist approach to behavior and biology in schizophrenia. *Psychological Review*, 99(1), 45-77.
- Cooper, R., & Shallice, T. (2000). Contention scheduling and the control of routine activities. *Cognitive Neuropsychology*, 17(4), 297-338.
- Creem, S. H., & Proffitt, D. R. (2001). Grasping objects by their handles: A necessary interaction between cognition and action. *Journal of Experimental Psychology: Human Perception and Performance*, 27(1), 218-228.
- Dell, G. S., Berger, L. K., & Svec, W. R. (1997). Language production and serial order: A functional analysis and a model. *Psychological Review*, 104(1), 123-147.
- Dell, G. S., Chang, F., & Griffin, Z. M. (1999). Connectionist models of language production: Lexical access and grammatical encoding. *Cognitive Science*, 23(4), 517-542.
- Dell, G. S., Juliano, C., & Govindjee, A. (1993). Structure and content in language production: A theory of frame constraints in phonological speech errors. *Cognitive Science*, 17, 149-95.
- Della Malva, C. L., Stuss, D. T., D'Alton, J., & Willmer, J. (1993). Capture errors and sequencing after frontal brain lesions. *Neuropsychologia*, 31(4), 363-372.
- DellaSala, S., Marchetti, C., & Spinnler, H. (1994). The anarchic hand: A fronto-mesial sign. In J. Boller & J. Grafman (Eds.), *Handbook of neuropsychology*. Amsterdam: Elsevier.
- DeRenzi, E., & Lucchelli, F. (1988). Ideational apraxia. *Brain*, 111, 1173-1185.
- Duncan, J. (1986). Disorganization of behavior after frontal lobe damage. *Cognitive Neuropsychology*, 3, 271-290.
- Duncan, J. (1993). Selection of input and goal in control of behavior. In A. Baddeley & L. Weiskrantz (Eds.), *Attention: Selection, awareness, and control* (p. 53-71). New York: Clarendon.
- Duncan, J. (1996). Cooperating brain systems in selective perception and action. In J. L. McClelland & T. Inui (Eds.), *Attention and performance 16: Information integration in perception and communication* (p. 549-578). Cambridge, MA: MIT Press.
- Elman, J. L. (1990). Finding structure in time. *Cognitive Science*, 14, 179-211.
- Elman, J. L. (1991). Distributed representations, simple recurrent networks, and grammatical structure. *Machine Learning*, 7, 195-225.
- Elman, J. L. (1993). Learning and development in neural networks: The importance of starting small. *Cognition*, 48, 71-99.
- Elsner, B., & Hommel, B. (2001). Effect anticipation and action control. *Journal of Experimental Psychology: Human Perception and Performance*, 27(1), 229-240.
- Ericsson, K. A., & Kintsch, W. (1995). Long-term working memory. *Psychological Review*, 102(2), 211-245.
- Eriksen, B. W., & Eriksen, C. W. (1974). Effects of noise letters upon the identification of a target letter in a nonsearch task. *Perception and Psychophysics*, 16, 143-149.
- Estes, W. K. (1972). An associative basis for coding and organization in memory. In A. W. Melton & M. E. (Eds.), *Coding processes in human memory*. New York: Halsted Press.

- Fuster, J. (1995). Temporal processing. In J. Grafman, K. J. Holyoak, & F. Boller (Eds.), *Structure and functions of the human prefrontal cortex* (p. 173-181). New York: New York Academy of Sciences.
- Fuster, J. M. (1989). *The prefrontal cortex*. New York: Raven Press.
- Godbout, L., & Doyon, J. (1995). Mental representations of knowledge following frontal-lobe or postrolandic lesions. *Neuropsychologia*, 33, 1671-1696.
- Grafman, J. (1995). Similarities and differences among current models of prefrontal cortical functions. In J. Grafman, K. J. Holyoak, & F. Boller (Eds.), *Structure and functions of the human prefrontal cortex* (p. 337-368). New York: New York Academy of Sciences.
- Greenfield, P. M. (1977). Building a tree structure: The development of hierarchical complexity and interrupted strategies in children's construction activity. *Developmental Psychology*, 13(4), 299-313.
- Grossberg, S. (1986). The adaptive self-organization of serial order in behavior: Speech, language, and motor control. In E. C. Schwab & H. C. Nusbaum (Eds.), *Pattern recognition by humans and machines, vol. 1: Speech perception* (p. 187-294). New York: Academic Press.
- Gupta, P., & Dell, G. S. (1999). The emergence of language from serial order and procedural memory. In B. MacWhinney (Ed.), *The emergence of language* (p. 447-481). Mahwah, New Jersey: Lawrence Erlbaum Associates.
- Hanson, C., & Hanson, S. J. (1996). Development of schemata during event parsing: Neisser's perceptual cycle as a recurrent connectionist network. *Journal of Cognitive Neuroscience*, 8(2), 119-134.
- Hartley, T. A., & Houghton, G. (1996). A linguistically constrained model of short-term memory for nonwords. *Journal of Memory and Language*, 35, 1-31.
- Hayhoe, M. (2000). Vision using routines: A functional account of vision. *Visual Cognition*, 7, 43-64.
- Hikosaka, O., Nakahara, H., Rand, M. K., Sakai, K., Lu, X., Nakamura, K., Miyachi, S., & Doya, K. (1999). Parallel neural networks for learning sequential procedures. *Trends in Neurosciences*, 22(12), 464-71.
- Hinton, G. E. (1989). Connectionist learning procedures. *Artificial Intelligence*, 40, 185-234.
- Hinton, G. E., McClelland, J. L., & Rumelhart, D. E. (1986). Distributed representations. In D. E. Rumelhart & J. L. McClelland (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition. volume 1: Foundations* (p. 77-109). Cambridge, Massachusetts: MIT Press.
- Houghton, G. (1990). The problem of serial order: A neural network model of sequence learning and recall. In R. Dale, C. Mellish, & M. Zock (Eds.), *Current research in natural language generation*. San Diego, California: Academic Press.
- Houghton, G. (1994). Inhibitory control of neurodynamics. In M. Oaksford & G. D. A. Brown (Eds.), *Neurodynamics and psychology*. London: Academic Press.
- Houghton, G., & Hartley, T. (1995). Parallel models of serial behavior: Lashley revisited. *Psyche*, 2(25), 1-25.
- Humphreys, G. W., & Forde, E. M. E. (1999). Disordered action schema and action disorganization syndrome. *Cognitive Neuropsychology*, 15, 771-811.
- Humphreys, G. W., Forde, E. M. E., & Francis, D. (in press). The organization of sequential actions. In S. Monsell & J. Driver (Eds.), *Attention and performance XVIII*. Cambridge, MA: MIT Press.
- Humphreys, G. W., & Riddoch, M. J. (2001). Detection by action: Neuropsychological evidence for action-defined templates in search. *Nature Neuroscience*, 4(1), 84-88.
- James, W. (1890). *The principle of psychology*. New York: Holt.
- Jastrow, J. (1905). The lapses of consciousness. *Popular Science Monthly*, 67, 481-502.
- Jordan, M. I. (1986). *Serial order: A parallel distributed processing approach* (Tech. Rep. No. 8604). University of California, San Diego, Institute for Cognitive Science.
- Jordan, M. I. (1994). Attractor dynamics and parallelism in a connectionist sequential machine. In *Proceedings of the eighth annual conference of the cognitive science society* (p. 531-546). Englewood Cliffs, NJ: Erlbaum.
- Jordan, M. I., & Rumelhart, D. E. (1992). Forward models: Supervised learning with a distal teacher. *Cognitive Science*, 16, 307-354.
- Klatzky, R. L., Pellegrino, J., McCloskey, B. P., & Lederman, S. J. (1993). Cognitive representations of functional interactions with objects. *Memory and Cognition*, 21(3), 294-303.
- Kosslyn, S. (1994). *Image and brain*. Cambridge, MA: MIT Press.
- Kruskal, J. B., & Wish, M. (1978). *Multidimensional scaling*. Beverly Hills, CA: Sage University Series.
- Laird, J. E., Newell, A., & Rosenbloom, P. S. (1987). Soar: An architecture for general intelligence. *Artificial Intelligence*, 33, 1-64.
- Land, M. F., Mennie, N., & Rusted, J. (1999). Eye movements and the roles of vision in activities of daily living: making a cup of tea. *Perception*, 28, 1311-1328.
- Lashley, K. S. (1951). The problem of serial order in behavior. In L. A. Jeffress (Ed.), *Cerebral mechanisms in behavior: The hixon symposium*. New York: John Wiley and Sons.
- Lebiere, C., Anderson, J. R., & Reder, L. M. (1994). Error modeling in the act-r production system. In *Proceedings of the 16th annual conference of the cognitive science society*. Erlbaum.
- Lehmkuhl, G., & Poeck, K. (1981). A disturbance in the conceptual organization of actions in patients with ideational apraxia. *Cortex*, 17, 153-158.
- Lhermitte, F. (1983). Utilization behavior and its relation to lesions of the frontal lobes. *Brain*, 106, 237-55.
- Liberman, A. M., Cooper, F. S., Sankweiler, D. S., & Studdert-Kenedy, M. (1967). Perception of the speech code. *Psychological Review*, 74, 431-461.
- Luria, A. R. (1966). *Higher cortical functions in man*. New York: Basic Books.
- MacCleod, C. M. (1991). Half a century of research on the stroop effect: An integrative review. *Psychological Bulletin*, 109, 163-203.
- MacKay, D. G. (1985). A theory of the representation, organization, and timing of action, with implications for sequencing disorders. In E. A. Roy (Ed.), *Neuropsychological studies of apraxia* (p. 267-308). Amsterdam: Elsevier Science Publishers.
- MacKay, D. G. (1987). *The organization of perception and action: A theory for language and other cognitive skills*. New York: Springer-Verlag.
- McCallum, A. K. (1996). *Reinforcement learning with selective perception and hidden state*. Unpublished doctoral dissertation, University of Rochester, Department of Computer Science.

- McClelland, J. L., McNaughton, B. L., & O'Reilly, R. C. (1995). Why are there complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Psychological Review*, 419-457.
- McClelland, J. L., St. John, M., & Taraban, R. (1989). Sentence comprehension: A parallel distributed processing approach. *Language and Cognitive Processes*, 4, 287-335.
- Miller, G. A., Galanter, E., & Pribram, K. H. (1960). *Plans and the structure of behavior*. New York: Holt, Rinehart and Winston.
- Milner, B., & Petrides, M. (1984). Behavioural effects of frontal-lobe lesions in man. *Trends in Neurosciences*, 7(11), 403-407.
- Morgan, J. L., & Demuth, K. (1996). *Signal to syntax: Bootstrapping from speech to grammar in early acquisition*. Mahwah, NJ: Lawrence Erlbaum Associates.
- Mozer, M. C., & Sitton, M. (1996). Computational modeling of spatial attention. In H. Pashler (Ed.), *Attention*. Hove, UK: Psychology Press.
- Mueller, H. J., Humphreys, G. W., & Donnelly, N. (1994). Search via recursive rejection (SERR): Visual search for single and dual form-conjunction targets. *Journal of Experimental Psychology: Human Perception and Performance*, 20(2), 235-258.
- Newell, A. (1990). *Unified theories of cognition*. Cambridge, MA: MIT Press.
- Newell, A., & Simon, H. A. (1972). *Human problem solving*. Englewood Cliffs, NJ: Prentice-Hall.
- Norman, D. A. (1981). Categorization of action slips. *Psychological Review*, 88(1), 1-14.
- Norman, D. A., & Bobrow, D. G. (1979). Descriptions: An intermediate stage in memory retrieval. *Cognitive Psychology*, 11, 107-123.
- Norman, D. A., & Shallice, T. (1986). Attention to action: Willed and automatic control of behavior. In R. J. Davidson, G. E. Schwartz, & D. Shapiro (Eds.), *Consciousness and self-regulation: Advances in research and theory*. New York: Plenum Press.
- Perlmutter, B. (1989). Learning state-space trajectories. *Neural Computation*, 1(2), 263-269.
- Plaut, D. C., & Kello, C. T. (1999). The emergence of phonology from the interplay of speech comprehension and production: A distributed connectionist approach. In B. MacWhinney (Ed.), *The emergence of language* (p. 381-415). Mahway, N.J.: Erlbaum.
- Plaut, D. C., & McClelland, J. M. (2000). Stipulating vs. discovering representations. *Behavioral and Brain Sciences*, 23.
- Pylyshn, Z. (1989). The role of location indexes in spatial perception: A sketch of the first spatial-index model. *Cognition*, 32, 65-97.
- Reason, J. (1979). Actions not as planned: The price of automatization. In G. Underwood & R. Stevens (Eds.), *Aspects of consciousness, vol. 1: Psychological issues*. London: Academic Press.
- Reason, J. (1990). *Human error*. Cambridge: Cambridge University Press.
- Reason, J. T. (1984a). Absent-mindedness and cognitive control. In J. E. Harris & P. E. Morris (Eds.), *Everyday memory, actions, and absent-mindedness* (p. 113-132). London: Academic Press.
- Reason, J. T. (1984b). Lapses of attention in everyday life. In R. Parasuraman & D. D. R. (Eds.), *Varieties of attention* (p. 515-549). London: Academic Press.
- Reason, J. T. (1992). Cognitive underspecification: Its varieties and consequences. In B. J. Baars (Ed.), *Experimental slips and human error: Exploring the architecture of volition* (p. 71-91). New York: Plenum Press.
- Riddoch, M. J., Edwards, M. G., Humphreys, G. W., West, R., & Heafield, T. (1998). Visual affordances direct action: Neuropsychological evidence from manual interference. *Cognitive Neuropsychology*, 15, 645-683.
- Riddoch, M. J., Humphreys, G. W., & Edwards, M. G. (in press). Visual affordances and object selection. In S. Monsell & J. Driver (Eds.), *Attention and performance xviii*. Cambridge, MA: MIT Press.
- Rizzolatti, G., Craighero, L., & Sieroff, E. (1998). From spatial attention to attention to objects: An extension of the premotor theory of attention. *Revue de Neuropsychologie*, 8(1), 155-174.
- Rogers, T. T., & McClelland, J. L. (submitted). *Semantics without categorization: A parallel distributed processing approach to the acquisition and use of natural semantic knowledge*. (Manuscript submitted for publication.)
- Rohde, D. (1999). *Lens: The light, efficient network simulator*. Technical Report CMU-CS-99-164, Department of Computer Science, Carnegie Mellon University, Pittsburgh PA. (Available at <http://www.cs.cmu.edu/~dr/Lens/>)
- Rohde, D. L. T., & Plaut, D. C. (1999). Language acquisition in the absence of explicit negative evidence: How important is starting small? *Cognition*, 72(1), 67-109.
- Rosch, E., Mervis, C. B., Gray, W. D., Johnson, D. M., & Boyes-Braem, P. (1976). Basic objects in natural categories. *Cognitive Psychology*, 8, 382-439.
- Roy, E. A. (1982). Action and performance. In A. W. Ellis (Ed.), *Normality and pathology in cognitive functions* (p. 265-298). London: Academic Press.
- Rumelhart, D. A., & McClelland, J. L. (1986). *Parallel distributed processing: Explorations in the microstructure of cognition* (Vol. 1). Cambridge, MA: MIT Press.
- Rumelhart, D. E., Durbin, R., Golden, R., & Chauvin, Y. (1996). Backpropagation: The basic theory. In P. Smolensky, M. C. Mozer, & D. E. Rumelhart (Eds.), *Mathematical perspectives on neural networks* (p. 533-566). Hillsdale, NJ: Lawrence Erlbaum.
- Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representations by error propagation. In D. E. Rumelhart & J. L. McClelland (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition. volume 2: Psychological and biological models* (Vol. 1, p. 318-362). Cambridge, Massachusetts: MIT Press.
- Rumelhart, D. E., & Norman, D. A. (1982). Simulating a skilled typist: A study of skilled cognitive-motor performance. *Cognitive Science*, 6, 1-36.
- Rumelhart, D. E., Smolensky, P., McClelland, J. L., & Hinton, G. E. (1986). Schemata and sequential thought processes in pdp models. In J. L. McClelland & D. E. Rumelhart (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition. volume 2: Psychological and biological models* (p. 7-57). Cambridge, Massachusetts: MIT Press.
- Schank, R. C. (1982). *Dynamic memory: A theory of reminding and learning in computers and people*. New York: Cambridge University Press.
- Schank, R. C., & Abelson, R. P. (1977). *Scripts, plans, goals and understanding*. Hillsdale, NJ: Lawrence Erlbaum Associates.
- Schwartz, M. F. (1995). Re-examining the role of executive functions in routine action production. In J. Grafman, K. J. Holyoak,

- & F. Boller (Eds.), *Structure and functions of the human pre-frontal cortex* (p. 321-335). New York: New York Academy of Sciences.
- Schwartz, M. F., & Buxbaum, L. J. (1997). Naturalistic action. In K. M. Heilman (Ed.), *Apraxia: The neuropsychology of action* (p. 269-289). Hove, England: Psychology Press.
- Schwartz, M. F., Mayer, N. H., Fitzpatrick, E. J., & Montgomery, M. W. (1993). Cognitive theory and the study of everyday action disorders after brain damage. *Journal of Head Trauma Rehabilitation*, 8, 59-72.
- Schwartz, M. F., Montgomery, M. W., Buxbaum, L. J., Lee, S. S., Carew, T. G., Coslett, H. B., Ferraro, M., & Fitzpatrick-DeSalme, E. J. (1998). Naturalistic action impairment in closed head injury. *Neuropsychology*, 12(1), 13-28.
- Schwartz, M. F., Montgomery, M. W., Fitzpatrick-DeSalme, E. J., Ochipa, C., Coslett, H. B., & Mayer, N. H. (1995). Analysis of a disorder of everyday action. *Cognitive Neuropsychology*, 8, 863-892.
- Schwartz, M. F., Reed, E. S., Montgomery, M. W., Palmer, C., & Mayer, N. H. (1991). The quantitative description of action disorganization after brain damage: A case study. *Cognitive Neuropsychology*, 8(5), 381-414.
- Servan-Schreiber, D., Cleeremans, A., & McClelland, J. L. (1988). *Encoding sequential structure in simple recurrent networks* (Tech. Rep. No. CMU-CS-183). Carnegie Mellon University, Department of Computer Science.
- Servan-Schreiber, D., Cleeremans, A., & McClelland, J. L. (1991). Graded state machines: The representation of temporal contingencies in simple recurrent networks. *Machine Learning*, 7, 161-193.
- Shallice, T. (1988). *From neuropsychology to mental structure*. Cambridge: Cambridge University Press.
- Shallice, T., Burgess, P. W., Schon, F., & Baxter, D. M. (1989). The origins of utilization behavior. *Brain*, 112, 1587-1598.
- Stadler, M. A., & Frensch, P. A. (1998). *Handbook of implicit learning*. Thousand Oaks, CA: Sage Publications.
- Treisman, A. (1988). Features and objects: The fourteenth Bartlett memorial lecture. *Quarterly Journal of Experimental Psychology*, 40, 201-237.
- Tucker, M., & Ellis, R. (1998). On the relations between seen objects and components of potential actions. *Journal of Experimental Psychology: Human Perception and Performance*, 24(3), 830-846.
- Ullman, S. (1984). Visual routines. *Cognition*, 18, 97-157.
- Van Lehn, K. (1989). Problem solving and skill acquisition. In M. Posner (Ed.), *Foundations of cognitive science* (p. 526-79). Cambridge, MA: MIT Press.
- Van Lehn, K., Ball, W., & Kowalski, B. (1989). Non-lifo execution of cognitive procedures. *Cognitive Science*, 13, 415-465.
- Wall, L., Christiansen, T., & Orwant, J. (2000). *Programming Perl* (3rd Edition). Cambridge, MA: O'Reilly & Associates.
- Whitehead, S. D., & Ballard, D. H. (1990). Active perception and reinforcement learning. *Neural Computation*, 2, 409-419.
- Whitehead, S. D., & Lin, L. (1995). Reinforcement learning of non-markov decision processes. *Artificial Intelligence*, 73, 271-305.
- Williams, R. J., & Zipser, D. (1995). Gradient-based learning algorithms for recurrent networks and their computational complexity. In Y. Chauvin & D. E. Rumelhart (Eds.), *Back-propagation: Theory, architectures and applications* (p. 433-486). Hillsdale, NJ: Erlbaum.
- Zacks, J. M., & Tversky, B. (2001). Event structure in perception and conception. *Psychological Bulletin*, 127, 3-21.