Error Modeling in the ACT-R Production System

Christian Lebière
Department of Psychology
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213
cl+@cmu.edu

John R. Anderson
Department of Psychology
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213
ja+@cmu.edu

Lynne M. Reder
Department of Psychology
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213
reder+@cmu.edu

Abstract

We describe how to extend the ACT-R production system to model human errors in the performance of a high-level cognitive task: to solve simple linear algebra problems while memorizing a digit span. Errors of omission are produced by introducing a cutoff on the latency of memory retrievals. If a memory chunk cannot gather enough activation to be retrieved before the threshold is reached, retrieval fails. Adding Gaussian noise to chunk activation produces a pattern quantitatively similar to subject errors. Errors of commission are introduced by allowing imperfect matching in the condition side of productions. The wrong memory chunk can be retrieved if its activation is large enough to allow it to overcome the mismatch penalty. This mechanism provides a qualitative and quantitative fit to subject errors. In conclusion, this paper demonstrates that human-like errors, sometimes thought of as the exclusive domain of connectionist models, can be successfully duplicated in production system models.

Introduction

ACT-R (Anderson, 1993) is a model of human cognition which assumes that a production system operates on a declarative memory. It is a successor to previous production system models (Anderson, 1976, 1983) and continues the emphasis on activation-based processes as the mechanism for relating the production system to the declarative memory. Different declarative memories have different levels of activation which determine their rate and probability of processing by the production rules. ACT-R is distinguished from the prior ACT theories in that the details of its design have been strongly guided by the rational analysis of Anderson (1990). Essentially it is a production system tuned to achieve optimal performance given the statistical structure of the environment.

Errors are a fundamental aspect of human cognition that symbolic systems have never been able to completely model. Symbolic systems have been able to model consistent errors by assuming that people have systematic bugs (Van Lehn, 1989) and errors of commission by assuming certain rules fail to apply. They have much more difficulty with the occasional slips or intrusions (Norman, 1981). These systems are sometimes thought of as too precise, too deterministic and too algorithmic to be able to exhibit the random, gradual degradation of performance exhibited by humans. A symbolic system works or it does not. When connectionist models became popular (Rumelhart and McClelland, 1986), one of their main attractions was that their holistic computation style could exhibit a capacity for human-like errors and graceful degradation of performance under noise or component failures. ACT-R is a hybrid system. Although its declarative elements are symbolic structures and its procedural rules implement an algorithmic matching process, the activation of the chunks is spread through a connection network. In this paper, we describe how to model errors with the ACT-R system by redefining (part of) the matching process as an activation-based constraint-satisfaction mechanism.

To be concrete, we will explain this with respect to the following task which is described more fully in Anderson, Reder, & Lebière (in preparation). Subjects were asked to memorize a digit span of 2, 4 or 6 digits, and then solve a linear equation before recalling the digits. The equation types are detailed in Table 1. Types (1) through impacted on the correctly. After

Table 2

```
p solve_x/a=b
   =goal>
        isa   solve-equation
        lht   =term
        rht   =b
   =term>
        isa   term
        lho   x
        op    /
        rho   =rho
==>
   =product>
        isa   term
        lho   =b
        op    *
        rho   =rho
   =goal>
        lht   x
        rht   =product
```

```
p compute_a*b
   =goal>
        isa   solve-equation
        lht   x
        rht   =term
   =term>
        isa   term
        lho   =lho
        op    *
        rho   =rho
   =lho*rho>
        isa   times-fact
        arg1  =lho
        arg2  =rho
        product  =product
==>
   =goal>
        rht   =product
```

```
p output_digit
   =goal>
        isa   recall-span
        pointer  =index
   =item>
        isa   memory-span
        item  =value
        position  =index
        next  =next
==>
   =goal>
        pointer  =next
```

Basic Model

An ACT-R model of this task would contain three sets of productions: one to manipulate the equations, one to compute arithmetic solutions, and one to retrieve digits from the span. Simplified examples of each production set are included in Table 2. ACT-R productions are composed of the keyword '**p**', the name of the production, then a number of chunk retrievals (the left-hand side)

between them, called **Interactive Association (IA)**^{2}. Thus, as an example, faced with the equation x=3*4, the chunk encoding 3*4=12 will receive activation to the degree that 3 and 4 are active as sources and to the degree they are strongly associated to the chunk.

The level of activation of a chunk determines not only its probability of retrieval but also the retrieval time. The latency is defined as inversely proportional to the exponential of the chunk activation (and the production strength). This means that direct retrievals, which gather high levels of activation, will take a relatively short time whereas indirect retrievals will take increasingly long as their activation level decreases. It seems natural to impose a cutoff time on how long a retrieval can take. If a chunk cannot gather enough activation, its retrieval will exceed the **latency threshold** and will fail. To introduce some randomness in this deterministic behavior, it is necessary to add some noise to the chunk activations. This Gaussian noise mechanism is a standard ACT-R feature. Increasing the noise or decreasing the activation threshold will increase the number of errors. At equal error frequencies, the former will increase the randomness of the errors while the latter will decrease it.

Finally, to ensure that this threshold applies equally in all situations, it is necessary to make one additional assumption regarding activation: that the total source level

The errors in the digit span task also were largely errors of commission. Some of these errors were systematic. Subjects showed a tendency to recall a digit from a near position in the span producing a generalization effect (Nairne, 1992). Also, digits were not equally frequent and subjects showed a tendency to intrude the more frequent digits.

Mismatch Penalty and Errors of Commission

Allowing more active but only partially matching chunks to be retrieved instead of the correct ones can account for these errors of commission. Partial matching has received support, both empirically and computationally in recent work of Reder (Reder & Cleeremans, 1990; Reder & Kusbit, 1991; Reder & Ritter, 1992; Reder, Richards & Stroffolino, in prep.). We changed ACT-R's pattern-matcher so that rather than retrieving all combinations of chunks that matched the condition of a production, it selected the most active chunk. Chunks accumulate activation from sources in the environment. If there is a mismatch of a chunk to a production pattern, its activation level will be reduced by a mismatch penalty. A mismatch takes the form of a contradiction between what is required by a slot value in the pattern and the actual slot value of the matching chunk. For instance, the pattern may be looking for 3+4 and the chunk 3+5=8 will have 5 where 4 is required. The mismatch penalty is proportional to the degree of mismatch between the desired and actual slot values. The IA values between desired chunks and the actual chunks are interpreted as their degree of similarity.

Let us first see how this mechanism can account for the pattern of arithmetic errors. When retrieving the sum of 2 and 5, both numbers are made sources and contribute activation to the correct fact: 2+5=7. Many other facts also receive activation from these and other numbers and might even gather more activation than the desired fact because of the pattern of sources and the Gaussian noise in the activation values. To favor close matches, IA values between numbers are set to reflect their similarity.[4] Therefore, the penalty for 2+6=8 will be less than the penalty for 2+1=3, because 6 is more similar to 5 than 1. All things being equal, the former will be more active than the latter and will have a better chance of being retrieved (if a misretrieval occurs), which explains the predominance of close matches.

To model transformation errors, we need to switch from the model described in a previous section where each transformation rule was encoded by its own production and applied without indirect retrieval to one with only one production retrieving a different algebraic rule for each transformation. A simplified version of such a production is shown in the left-hand side of Table 4.

Essentially, the production retrieves by an indirect match the declarative rule which best matches the left-hand term of the equation, then constructs a new right-hand term using the new operator provided by the rule. Sample

Table 4

p transform	Chunks encoding rules:
=goal>	
isa solve-equation	x+a-rule
lht =term	isa rule
rht =b	lho x
=term>	op +
isa term	rho a
lho =lho	new-op -
op =op	
rho =rho	x-a-rule
=rule>	isa rule
isa rule	lho x
lho =lho	op -
op =op	rho a
rho =rho	new-op +
new-op =new-op	
==>	x*a-rule
=new-term>	isa rule
isa term	lho x
lho =b	op *
op =new-op	rho a
rho =rho	new-op /
=goal>	
lht =lho	...
rht =new-term	

[4] Specifically, IA(i,j) = exp(-|i-j|).