Introduction to ACT-R 6.0

Tutorial

Frank Ritter
Penn State University
Applied Cognition Laboratory
ritter@ist.psu.edu

Mike Schoelles
Rensselaer Polytechnic Institute
CogWorks Laboratory
schoem@rpi.edu

Some slides pinched from Lebiere
Tutorial Overview

- Cognitive Architecture/Modeling Overview
- ACT-R Theory Overview
  - Addition, counting and letter models
- Build a model (Dialing Model)
- ACT-R Theory Details
  - Sternberg and Building Sticks models
- Current Directions
What is a Cognitive Architecture?

- Infrastructure for an intelligent system
- Cognitive functions that are constant over time and across different task domains
- Analogous to a building, car, or computer
Unified Theories of Cognition

- Account of intelligent behavior at the system-level
- Newell’s claim
  - “You can’t play 20 questions with nature and win”
Integrated Cognitive Architecture

- Cognition doesn’t function in isolation
  - Interaction with perception, motor, auditory, etc. systems
- Embodied cognition
  - Represents a shift from
    - “mind as an abstract information processing system”
    - Perceptual and motor are merely input and output systems
  - Must consider the role of the environment
Motivations for a Cognitive Architecture

1. **Philosophy:** Provide a unified understanding of the mind.

2. **Psychology:** Account for experimental data.

3. **Education:** Provide cognitive models for intelligent tutoring systems and other learning environments.

4. **Human Computer Interaction:** Evaluate artifacts and help in their design.

5. **Computer Generated Forces:** Provide cognitive agents to inhabit training environments and games.

6. **Neuroscience:** Provide a framework for interpreting data from brain imaging.
Requirements for Cognitive Architectures

1. Integration, not just of different aspects of higher level cognition but of cognition, perception, and action.

2. Systems that run in real time.

3. Robust behavior in the face of error, the unexpected, and the unknown.

4. Parameter-free predictions of behavior.

5. Learning.
# Newell’s Time Scale of Human Activity (amended)

<table>
<thead>
<tr>
<th>Scale (sec)</th>
<th>Time Units</th>
<th>System</th>
<th>Analysis</th>
<th>World (theory)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100000</td>
<td>days</td>
<td>Task</td>
<td>Traditional Task Analysis</td>
<td>Bounded Rationality</td>
</tr>
<tr>
<td>10000</td>
<td>hours</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1000</td>
<td>10 min</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>min</td>
<td>Subtask</td>
<td>Unit Task Analysis</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>10 sec</td>
<td>Unit task</td>
<td>Cognitive Task Analysis</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 sec</td>
<td>Microstrategies (Activities)</td>
<td>Embodied Activities</td>
<td>Cognitive Band (symbolic)</td>
</tr>
<tr>
<td></td>
<td>1/3 sec</td>
<td>Embodied Activities (Basic units of embodied cognition)</td>
<td>Production Rules</td>
<td></td>
</tr>
<tr>
<td>0.1</td>
<td>100 ms</td>
<td>Production Rule (CPM-GOMS Operator)</td>
<td>Memory Elements</td>
<td></td>
</tr>
<tr>
<td>0.01</td>
<td>10 ms</td>
<td>Elements</td>
<td>Architectural</td>
<td>Biological Band (subsymbolic)</td>
</tr>
<tr>
<td>0.001</td>
<td>1 ms</td>
<td>Parameters</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Taxonomy

Computational Cognitive Models

- Connectionist
- Symbolic
- Mathematical

Cognitive Architectures
- Other AI
- Production System

Other AI
- Symbolic only
- SOAR
- EPIC
- Hybrid

ACT-R 5.0
Other Cognitive Architectures

- **Soar**
  - Production rule system
    - Organized in terms of operators associated with problem spaces
    - Goal oriented
      - Sub-goaling
    - Learning mechanism - Chunking

- **EPIC**
  - Parallel firing of production rules
  - Well developed visual and motor system
ACT-R Overview

- Modular
- Knowledge Representation
- Symbolic/Sub-symbolic
- Performance/Learning
Approach: Integrated Cognitive Models

- Cognitive model = computational process that thinks/acts like a person
- Integrated cognitive models...
Study 1: Lateral Deviation

- Deviation from lane center (RMSE)

Model Predictions

Human Data

![Diagram showing lateral deviation for different conditions: No Dialing, Full Manual, Speed Manual, Full Voice, Speed Voice. The diagram compares model predictions with human data.]
Study 1: Lateral Deviation

▼ Deviation from lane center (RMSE)

Model Predictions

Human Data
## History of the ACT-framework

<table>
<thead>
<tr>
<th>Predecessor</th>
<th>HAM</th>
<th>(Anderson &amp; Bower 1973)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theory versions</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACT-E</td>
<td>(Anderson, 1976)</td>
</tr>
<tr>
<td></td>
<td>ACT*</td>
<td>(Anderson, 1978)</td>
</tr>
<tr>
<td></td>
<td>ACT-R</td>
<td>(Anderson, 1993)</td>
</tr>
<tr>
<td></td>
<td>ACT-R 4.0</td>
<td>(Anderson &amp; Lebiere, 1998)</td>
</tr>
<tr>
<td></td>
<td>ACT-R 5.0</td>
<td>(Anderson &amp; Lebiere, 2001)</td>
</tr>
<tr>
<td></td>
<td>ACT-R 6</td>
<td>(Anderson et al., 2003, Anderson, 2007)</td>
</tr>
<tr>
<td>Implementations</td>
<td>GRAPES</td>
<td>(Sauers &amp; Farrell, 1982)</td>
</tr>
<tr>
<td></td>
<td>PUPS</td>
<td>(Anderson &amp; Thompson, 1989)</td>
</tr>
<tr>
<td></td>
<td>ACT-R 2.0</td>
<td>(Lebiere &amp; Kushmerick, 1993)</td>
</tr>
<tr>
<td></td>
<td>ACT-R 3.0</td>
<td>(Lebiere, 1995)</td>
</tr>
<tr>
<td></td>
<td>ACT-R 4.0</td>
<td>(Lebiere, 1998)</td>
</tr>
<tr>
<td></td>
<td>ACT-R/PM</td>
<td>(Byrne, 1998)</td>
</tr>
<tr>
<td></td>
<td>ACT-R 5.0</td>
<td>(Lebiere, 2001)</td>
</tr>
<tr>
<td></td>
<td>Windows Environment</td>
<td>(Bothell, 2001)</td>
</tr>
<tr>
<td></td>
<td>Macintosh Environment</td>
<td>(Fincham, 2001)</td>
</tr>
</tbody>
</table>
I. Perception & Attention
   1. Psychophysical Judgements
   2. Visual Search
   3. Eye Movements
   4. Psychological Refractory Period
   5. Task Switching
   6. Subitizing
   7. Stroop
   8. Driving Behavior
   9. Situational Awareness
  10. Graphical User Interfaces

II. Learning & Memory
   1. List Memory
   2. Fan Effect
   3. Implicit Learning
   4. Skill Acquisition
   5. Cognitive Arithmetic
   6. Category Learning
   7. Learning by Exploration and Demonstration
   8. Updating Memory & Prospective Memory
   9. Causal Learning

III. Problem Solving & Decision Making
   1. Tower of Hanoi
   2. Choice & Strategy Selection
   3. Mathematical Problem Solving
   4. Spatial Reasoning
   5. Dynamic Systems
   6. Use and Design of Artifacts
   7. Game Playing
   8. Insight and Scientific Discovery

IV. Language Processing
   1. Parsing
   2. Analogy & Metaphor
   3. Learning
   4. Sentence Memory

V. Other
   1. Cognitive Development
   2. Individual Differences
   3. Emotion
   4. Cognitive Workload
   5. Computer Generated Forces
   6. fMRI
   7. Communication, Negotiation, Group Decision Making

ACT-R 5.0 Architecture

- Current Goal
- Motor Modules
- Declarative Memory
- Perceptual Modules

Pathways:
- Current Goal → Test → Modify
- Motor Modules → Series of Actions
- Declarative Memory → Retrieve → Check
- Perceptual Modules → Identify Object
- Pattern Matching And Production Selection
- Schedule Action
- Move Attention
- Identify Object

Environment
ACT-R 6.0

Environment
# ACT-R: Assumption Space

<table>
<thead>
<tr>
<th>Performance</th>
<th>Declarative</th>
<th>Procedural</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Symbolic</strong></td>
<td>Retrieval of Chunks</td>
<td>Application of Production Rules</td>
</tr>
<tr>
<td><strong>Subsymbolic</strong></td>
<td>Noisy Activations Control Speed and Accuracy</td>
<td>Noisy Utilities Control Choice</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Learning</th>
<th>Declarative</th>
<th>Procedural</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Symbolic</strong></td>
<td>Encoding Environment and Caching Goals</td>
<td>Production Compilation</td>
</tr>
<tr>
<td><strong>Subsymbolic</strong></td>
<td>Bayesian Learning</td>
<td>Bayesian Learning</td>
</tr>
</tbody>
</table>
Interactive Session

- Load and run Addition model
  [actr6/tutorial/unit1/addition.lisp]
ACT-R: Knowledge Representation

Declarative-Procedural Distinction

Declarative Knowledge: Chunks
Configurations of small numbers of elements

Procedural Knowledge: Production Rules for retrieving chunks to solve problems.

336 +848
4

IF the goal is to add the numbers in a column and \( n1 + n2 \) are in the column
THEN retrieve the sum of \( n1 \) and \( n2 \).

Productions serve to coordinate the retrieval of information from declarative memory and the environment to produce transformations in the goal state.
Chunks: Example

\[
( \text{CHUNK-TYPE} \quad \text{NAME} \quad \text{SLOT1} \quad \text{SLOT2} \quad \text{SLOTN} )
\]

\[
( \text{FACT3+4} \\
\text{isa} \quad \text{ADDITION-FACT} \\
\text{ADDEND1} \quad \text{THREE} \\
\text{ADDEND2} \quad \text{FOUR} \\
\text{SUM} \quad \text{SEVEN} )
\]
Chunks: Example

(CLEAR-ALL)
(CHUNK-TYPE addition-fact addend1 addend2 sum)
(CHUNK-TYPE integer value)
(ADD-DM (fact3+4
  isa addition-fact
  addend1 three
  addend2 four
  sum seven)
  (three
    isa integer
    value 3)
  (four
    isa integer
    value 4)
  (seven
    isa integer
    value 7)
Chunks: Example

- **VALUE**
  - **THREE**
  - **ADDEND1**
  - **ADDEND2**
  - **FOUR**
    - **INTEGER**
  - **SUM**
    - **SEVEN**

- **ADDITION-FACT**
  - **isa**

**Examples of Values:**
- **3**
- **7**
- **4**

**Chunk**

**VALUE**

**INTEGER**
A Production is

1. The greatest idea in cognitive science.
2. The least appreciated construct in cognitive science.
3. A 50 millisecond step of cognition.
4. The source of the serial bottleneck in otherwise parallel system.
5. A condition-action data structure with “variables”.
6. A formal specification of the flow of information from cortex to basal ganglia and back again.
Productions

Key Properties

- modularity
- abstraction
- goal/buffer factoring
- conditional asymmetry

Structure of productions

(condition part

(delimiter

(action part

Specification of Buffer Tests

Specification of Buffer Transformations

( p name

==>

)
Interactive Session

- Load and run Counting model
The Modules

- Cognition
- Memory
- Vision
- Motor
- Audition
- Speech
ACT-R 6.0 Buffers

1. Goal Buffer (=goal, +goal)
   - represents where one is in the task
   - preserves information across production cycles
2. Retrieval Buffer (=retrieval, +retrieval)
   - holds information retrieval from declarative memory
   - seat of activation computations
3. Visual Buffers
   - location (=visual-location, +visual-location)
   - visual objects (=visual, +visual)
   - attention switch corresponds to buffer transformation
4. Auditory Buffers (=aural, +aural)
   - analogous to visual
   - elaborate theory of manual movement include feature preparation, Fitts law, and device properties
6. Vocal Buffers (=vocal, +vocal)
   - analogous to manual buffers but less well developed
7-N. Other buffers as needed, e.g., time
Cognition

- Executive Control - Production System
- Serial
- Parallel at sub-symbolic level
  - Utility selects production to fire
  - Utility = benefit - cost
    - Benefit = probability of success * value of achieving goal
Production System Cycle

- Match **conditions** of all rules to **buffers**
- Those that match enter the **conflict set**
- **Conflict resolution** selects a rule to **fire**
- **Action** side of rule initiates changes to one or more **buffers**
- If no **production** can match and no **action** is in progress then quit else repeat
Goal directed

- Represents what you are trying to do
- A declarative memory element that is the focus of “internal” attention
Memory Module

- Activation based
  - Frequency and recency
  - Contextual cues

- Cognition
  - Requests retrieval
    - Specifies constraints
    - Partial matching

- Memory
  - Parallel search of memory to match constraints
  - Calculates activation of matching chunks
  - Returns most active chunk
Vision Module

- ACT-R’s “eyes”
- Dorsal “where” system
- Ventral “what” system
“Where” System

- Cognition
  - Requests “pre-attentive” visual search
  - Specifies a set of constraints
  - Attribute/value pairs
    - Properties or spatial location
      - e.g. color red, screen-x greater-than 150

- “Where” system
  - Returns a “location” chunk
    - Specifies location of an object whose features satisfy the constraints

- Onsets
  - Features are held in vision module’s memory
Vision Module Memory
“What” System

- Cognition
  - Requests “move attention”
  - Provides “location” chunk

- “Where” System
  - Shifts visual attention to that location
  - Encodes object at that location
    - Added to Declarative Memory
    - Episodic representation of visual scene
  - Places encoding in “visual” buffer
  - Calculates latency
    - EMMA
Motor Module

- ACT-R’s Hands
- Based on EPIC’s Manual Motor Processor
- Movement Styles
- Phased Processing
Movement Styles

- Ply - moves a device (e.g., mouse) to a given location
- Punch - pressing a key below finger
- Peck - directed movement of finger to a location followed by keystroke
- Peck-recoil - same as peck but finger moves back
- Point-hand - moves hand to a new location
Phased Processing (1)

- Preparation Phase
  - Hierarchical feature preparation
    - Style->hand->finger
  - Prep time depends on
    - Complexity of movement
    - Number of features
  - State buffer set to prep busy
Phased Processing (2)

- Initiation (fixed 50 ms)
- Execution
  - *Time depends on*
    - Type of movement
      - Minimum execution time
    - Distance
      - Fitt’s Law
- Allow overlapping of preparation and execution
Interactive Session

- Load and run a model that uses PM buffers
Device Interface

- Simulated device with which ACT-R interacts
  - Contains graphical objects
- Typically a Window
  - Can be entire screen
- Interaction
  - Constructing vision system’s iconic memory (sets of features) from graphical objects
  - Handle mouse and keyboard actions
Audition Module

- Simulated perception of audio
- Memory of features
  - Temporal-extent - sound events
- Tones, digits, and speech
- Attributes
  - Onset, duration, delay, recode time
Audition Module Processing

- Parallels vision system
- Cognition
  - Specifies a set of constraints
  - Attribute/value pairs
- Audition
  - Returns a "location" chunk
- Cognition
  - Requests shift of auditory attention providing the "location" chunk
- Audition
  - Encodes the sound
Sub-symbolic level

- Sub-symbolic learning allow the system to adapt to the statistical structure of the environment.

- Production Utilities are responsible for determining which productions get selected when there is a conflict.

- Chunk Activations are responsible for determining which (if any chunks) get retrieved and how long it takes to retrieve them.

- Chunk Activations have been simplified in ACT-R 5.0 and a major step has been taken towards the goal of parameter-free predictions by fixing a number of the parameters.
Parameters

- Noise
  - Utility and activation

- Learning
  - Activation - frequency and recency
  - Utility - probability and cost

- Thresholds
  - Utility and activation
Build Dialing Model
Detailed ACT-R theory

- Activation equation
- Production Utility equation
- Production Compilation
Activation

\[ A_i = B_i + \sum_j W_j \cdot S_{ji} + \sum_k MP_k \cdot Sim_{kl} + N(0, s) \]

Conditions

Addend1 Three

Addend2 Four

Chunk i

Sum

=Goal>
isa write
relation sum
arg1 Three
arg2 Four

+Retrieval>
isa addition-fact
addend1 Three
addend2 Four

+/Retrieval>
isa addend1 Three
addend2 Four

Actions

Sim_{kl}

50
Activation makes chunks available to the degree that past experiences indicate that they will be useful at the particular moment:

Base-level: general past access/usefulness
Associative Activation: relevance to the general context
Matching Penalty: relevance to the specific match required
Noise: stochastic is useful to avoid getting stuck in local minima
Activation, Latency and Probability

• Retrieval time for a chunk is a negative exponential function of its activation:

\[ \text{Time}_i = F \cdot e^{-A_i} \]

• Probability of retrieval of a chunk follows the Boltzmann (softmax) distribution:

\[ t = \sqrt{2} \cdot s = \frac{\sqrt{6} \cdot \sigma}{\pi} \quad \quad P_i = \frac{e^{A_i}}{\sum_j e^{A_j}} \]

• The chunk with the highest activation is retrieved provided that it reaches the retrieval threshold \( \tau \)

• For purposes of latency and probability, the threshold can be considered as a virtual chunk
Base-level Activation

\[ A_i = B_i \]

The base level activation \( B_i \) of chunk \( C_i \) reflects a context-independent estimation of how likely \( C_i \) is to match a production, i.e. \( B_i \) is an estimate of the log odds that \( C_i \) will be used.

Two factors determine \( B_i \):
- frequency of using \( C_i \)
- recency with which \( C_i \) was used

\[ B_i = \ln \left( \frac{P(C_i)}{P(\overline{C_i})} \right) \]
Source Activation

\[ W = \sum W_j \] reflects an individual difference parameter.

The source activations \( W_j \) reflect the amount of attention given to elements, i.e. fillers, of the current goal. ACT-R assumes a fixed capacity for source activation.
The association strength $S_{ji}$ between chunks $C_j$ and $C_i$ is a measure of how often $C_i$ was needed (retrieved) when $C_j$ was element of the goal, i.e. $S_{ji}$ estimates the log likelihood ratio of $C_j$ being a source of activation if $C_i$ was retrieved.

$$S_{ji} = \ln \left( \frac{P(N_i|C_j)}{P(N_i)} \right)$$

$$= S - \ln(P(N_i|C_j))$$
Partial Matching

\[ + \left( \text{mismatch penalty} \times \text{similarity value} \right) \]

\[ + \sum_k MP_k \cdot Sim_{kl} \]

• The mismatch penalty is a measure of the amount of control over memory retrieval: \( MP = 0 \) is free association; \( MP \) very large means perfect matching; intermediate values allow some mismatching in search of a memory match.

• Similarity values between desired value \( k \) specified by the production and actual value \( l \) present in the retrieved chunk. This provides generalization properties similar to those in neural networks; the similarity value is essentially equivalent to the dot-product between distributed representations.
Noise

\[ + \text{ noise} \]

\[ + N(0, s) \]

- Noise provides the essential stochasticity of human behavior
- Noise also provides a powerful way of exploring the world
- Activation noise is composed of two noises:
  - A permanent noise accounting for encoding variability
  - A transient noise for moment-to-moment variation
Base-Level Learning

Based on the Rational Analysis of the Environment  
(Schooler & Anderson, 1997)

Base-Level Activation reflects the log-odds that a chunk will be needed. In the environment the odds that a fact will be needed decays as a power function of how long it has been since it has been used. The effects of multiple uses sum in determining the odds of being used.

\[
B_i = \ln\left(\sum_{j=1}^{n} t_j^{-d}\right)
\]

Base-Level Learning Equation

\[
B_i \approx \ln \left\{ \sum_{j=1}^{k} t_j^{-d} + \frac{n-k}{1-d} \frac{L^{1-d} - t_k^{1-d}}{L - t_k} \right\}
\]

\[
\approx n(n/(1-d)) - d*n(L)
\]

Note: The decay parameter d has been set to .5 in most ACT-R models
Interactive Session

- Load and run Sternberg model
Production Utility

Making Choices: Conflict Resolution

Expected Gain = \( E = PG - C \)  
\( P \) is expected probability of success  
\( G \) is value of goal  
\( C \) is expected cost

Probability of choosing \( i \) = \[ \frac{e^{E_i/t}}{\sum_j e^{E_j/t}} \]  
t reflects noise in evaluation and is like temperature in the Boltzmann equation

\( P = \frac{\text{Successes}}{\text{Successes} + \text{Failures}} \)  
a is prior successes  
m is experienced successes  
b is prior failures  
n is experienced failures

Successes = \( \alpha + m \)  
Failures = \( \beta + n \)
Decay of Experience

Successes(t) = \sum_{j=1}^{m} t_j^{-d}  

Success Discounting

Failures(t) = \sum_{j=1}^{n} t_j^{-d}  

Failure Discounting

Note: Such temporal weighting is critical in the real world.
Interactive Session

- Load and run Building Sticks model
Production Compilation: The Basic Idea

(p read-stimulus
 =goal>
   isa goal
   step attending
   state test
 =visual>
   isa text
   value =val
===>
 +retrieval>
   isa goal
   relation associate
   arg1 =val
   arg2 =ans

=goal>
   relation associate
   arg1 =val
   step testing)

(p recall-vanilla
 =goal>
   isa goal
   step attending
   state test
 =visual>
   isa text
   value "vanilla

===>
 +manual>
   isa press-key
   key =ans
   =goal>
   step waiting)

(p recall
 =goal>
   isa goal
   relation associate
   arg1 =val
   arg2 =ans

===>
 +manual>
   isa press-key
   key "7"
   =goal>
   relation associate
   arg1 "vanilla"
   step waiting)
Production Compilation: The Principles

1. Perceptual-Motor Buffers: Avoid compositions that will result in jamming when one tries to build two operations on the same buffer into the same production.

2. Retrieval Buffer: Except for failure tests proceduralize out and build more specific productions.


4. Safe Productions: Production will not produce any result that the original productions did not produce.

5. Parameter Setting:
Successes = P*initial-experience*
Failures = (1-P) *initial-experience*
Efforts = (Successes + Efforts)(C + *cost-penalty*)
Production Compilation: The Successes

1. Taatgen: Learning of inflection (English past and German plural). Shows that production compilation can come up with generalizations.

2. Taatgen: Learning of air-traffic control task - shows that production compilation can deal with complex perceptual motor skill.

3. Anderson: Learning of productions for performing paired associate task from instructions. Solves mystery of where the productions for doing an experiment come from.

4. Anderson: Learning to perform an anti-air warfare coordinator task from instructions. Shows the same as 2 & 3.

5. Anderson: Learning in the fan effect that produces the interaction between fan and practice. Justifies a major simplification in the parameterization of productions - no strength separate from utility.

Note all of these examples involve all forms of learning occurring in ACT-R simultaneous - acquiring new chunks, acquiring new productions, activation learning, and utility learning.
Predicting fMRI Bold Response from Buffer Activity

Example: Retrieval buffer during equation-solving predicts activity in left dorsolateral prefrontal cortex.

\[ BR(t) = \sum_i .344 D_i (t - t_i)^2 e^{-(t-t_i)^2/2} \]

where \( D_i \) is the duration of the \( i \)th retrieval and \( t_i \) is the time of initiation of the retrieval.
21 Second Structure of fMRI Trial

Load

Equation

c x + 3 = a

Blank Period

1.5 Second Scans

a=18
b=6
c=5
Solving $5x + 3 = 18$

Time 3.000: Find-Right-Term Selected
Time 3.050: Find-Right-Term Fired
Time 3.050: Module :VISION running command FIND- Time 3.050: Attend-Next-Term-Equation Selected
Time 3.100: Attend-Next-Term-Equation Fired
Time 3.100: Module :VISION running command MOVE- Time 3.150: Module :VISION running command FOCUS-ON
Time 3.150: Encode Selected
Time 3.200: Encode Fired
Time 3.281: 18 Retrieved
Time 3.281: Process-Value-Integer Selected
Time 3.331: Process-Value-Integer Fired
Time 3.331: Module :VISION running command FIND- Time 3.331: Attend-Next-Term-Equation Selected
Time 3.381: Attend-Next-Term-Equation Fired
Time 3.431: Encode Selected
Time 3.481: Encode Fired
Time 3.562: 3 Retrieved
Time 3.562: Process-Op1-Integer Selected
Time 3.612: Process-Op1-Integer Fired
Time 3.612: Module :VISION running command FIND- Time 3.612: Attend-Next-Term-Equation Selected
Time 3.662: Attend-Next-Term-Equation Fired
Time 3.712: Encode Selected
Time 3.762: Encode Fired
Time 4.362: Inverse-of--+ Retrieved
Time 4.362: Process-Operator Selected
Solving $5x + 3 = 18$ (cont.)

Time 4.412: Process-Operator Fired
Time 5.012: F318 Retrieved
Time 5.012: Finish-Operation1 Selected
Time 5.062: Finish-Operation1 Fired
Time 5.062: Module :VISION running command FIND- Time 5.062: Attend-Next-Term-Equation Selected
Time 5.112: Attend-Next-Term-Equation Fired
Time 5.112: Module :VISION running command MOVE-
Time 5.162: Module :VISION running command FOCUS-ON
Time 5.162: Encode Selected
Time 5.212: Encode Fired
Time 5.293: 5 Retrieved
Time 5.293: Process-Op2-Integer Selected
Time 5.343: Process-Op2-Integer Fired
Time 5.943: F315 Retrieved
Time 5.943: Finish-Operation2 Selected
Time 5.993: Finish-Operation2 Fired
Time 5.993: Retrieve-Key Selected
Time 6.043: Retrieve-Key Fired
Time 6.124: 3 Retrieved
Time 6.124: Generate-Answer Selected
Time 6.174: Generate-Answer Fired
Time 6.174: Module :MOTOR running command PRESS-KEY
Time 6.424: Module :MOTOR running command PREPARATION- Time 6.574: Device running command OUTPUT-KEY
("3" 3.574)
Left Dorsolateral Prefrontal Cortex
Bold Response for 2 Equation Types
Left Dorsolateral Prefrontal Cortex

Scan (1.5 sec.)

Percent Activation Change

- $\square - 5x + 3 = 18$
- $\square - cx + 3 = a$
Current Directions

ACT-R 6.0

Design goals

• More modular
• Consistent and uniform syntax
• Consistent treatment of buffers
• Parameter simplification

Model behavior in more complex real world environments
More Information

ACT-R Home Page: http://act.psy.cmu.edu