

CognitiveTutorAuthoringTools **LearnLab**
Pittsburgh Science of Learning Center

ACT-R

- theoretical and practical basis of the Cognitive Tutors, building on:
 - Lisp Tutor
 - Geometry Tutor
- production system
 - rule-based programming language
- Overview:
 - A simple example
 - A more complex example (addition)

1st Annual PSLC Summer School Building a cognitive model in Jess - 1 Pittsburgh, June 27 - July 1, 2005

CognitiveTutorAuthoringTools **LearnLab**
Pittsburgh Science of Learning Center

Components of a production rule system

- Working memory -- the database
- Production rule memory
- Interpreter that repeats the following cycle:
 - Match
 - Match "if-parts" of productions with working memory
 - Collect all applicable production rules
 - Conflict resolution
 - Select one of these productions to "fire"
 - Act
 - "Fire" production by making changes to working memory that are indicated in "then-part"

1st Annual PSLC Summer School Building a cognitive model in Jess - 2 Pittsburgh, June 27 - July 1, 2005

CognitiveTutorAuthoringTools **LearnLab**
Pittsburgh Science of Learning Center

An example production system

- You want a program that can answer questions and make inferences about food items
- Like:
 - What is purple and perishable?
 - What is packed in small containers and gives you a buzz?
 - What is green and weighs 15 lbs?

1st Annual PSLC Summer School Building a cognitive model in Jess - 3 Pittsburgh, June 27 - July 1, 2005

CognitiveTutorAuthoringTools **LearnLab**
Pittsburgh Science of Learning Center

A simple production rule system making inferences about food

WORKING MEMORY (WM)
Initially WM = (green, weighs-15-lbs)

RULE MEMORY

P1. **IF** green **THEN** produce
 P2. **IF** packed-in-small-container **THEN** delicacy
 P3. **IF** refrigerated **OR** produce **THEN** perishable
 P4. **IF** weighs-15-lbs **AND** inexpensive **AND NOT** perishable **THEN** staple
 P5. **IF** perishable **AND** weighs-15-lbs **THEN** turkey
 P6. **IF** weighs-15-lbs **AND** produce **THEN** watermelon

INTERPRETER

- Find all productions whose condition parts are true
- Deactivate productions that would add a duplicate symbol
- Execute the lowest numbered production (or quit)
- Repeat until there is no rule to execute

Adapted from the Handbook of AI, Vol I, pp. 191

1st Annual PSLC Summer School Building a cognitive model in Jess - 4 Pittsburgh, June 27 - July 1, 2005

CognitiveTutorAuthoringTools **LearnLab**
Pittsburgh Science of Learning Center

First cycle of execution

WORKING MEMORY
WM = (green, weighs-15-lbs)

CYCLE 1

- Productions whose condition parts are true: **P1**
- No production would add duplicate symbol
- Execute **P1**.
This gives: WM = (**produce**, green, weighs-15-lbs)

<p>RULE MEMORY</p> <p>P1. IF green THEN produce P2. IF packed-in-small-container THEN delicacy P3. IF refrigerated OR produce THEN perishable P4. IF weighs-15-lbs AND inexpensive AND NOT perishable THEN staple P5. IF perishable AND weighs-15-lbs THEN turkey P6. IF weighs-15-lbs AND produce THEN watermelon</p>	<p>INTERPRETER</p> <ol style="list-style-type: none"> Find all productions whose condition parts are true Deactivate productions that would add a duplicate symbol Execute the lowest numbered production (or quit) Repeat
---	---

Adapted from the Handbook of AI, Vol I, pp. 191

1st Annual PSLC Summer School Building a cognitive model in Jess - 6 Pittsburgh, June 27 - July 1, 2005

CognitiveTutorAuthoringTools **LearnLab**
Pittsburgh Science of Learning Center

Do this yourself before reading on!

- Hand simulate the execution of the production rule model.
- For each cycle, write down the following information:
 - Activate rules:
 - Deactivate rules:
 - Execute rule:
WM = (....)
- What is in working memory when the production rule model finishes?
- Are there any mistakes in the production rules?

1st Annual PSLC Summer School Building a cognitive model in Jess - 6 Pittsburgh, June 27 - July 1, 2005

Cycle 2

WORKING MEMORY

WM = (produce, green, weighs-15-lbs)

CYCLE 2

- Productions whose condition parts are true: **P1, P3, P6**
- Production P1 would add duplicate symbol, so **deactivate P1**
- Execute **P3** because it is the lowest numbered production.
This gives: WM = (**perishable**, produce, green, weighs-15-lbs)

RULE MEMORY

- P1. IF** green **THEN** produce
- P2. IF** packed-in-small-container **THEN** delicacy
- P3. IF** refrigerated **OR** produce **THEN** perishable
- P4. IF** weighs-15-lbs **AND** inexpensive **AND NOT** perishable **THEN** staple
- P5. IF** perishable **AND** weighs-15-lbs **THEN** turkey
- P6. IF** weighs-15-lbs **AND** produce **THEN** watermelon

INTERPRETER

- Find all productions whose condition parts are true
- Deactivate productions that would add a duplicate symbol
- Execute the lowest numbered production (or quit)
- Repeat

Adapted from the Handbook of AI, Vol 1, pp. 191

Cycle 3

WORKING MEMORY

WM = (perishable, produce, green, weighs-15-lbs)

CYCLE 3

- Productions whose condition parts are true: **P1, P3, P5, P6**
- Productions P1 and P3 would add duplicate symbol, so **deactivate P1 and P3**
- Execute **P5. Incorrect rule!?**
This gives: WM = (**turkey**, perishable, produce, green, weighs-15-lbs)

RULE MEMORY

- P1. IF** green **THEN** produce
- P2. IF** packed-in-small-container **THEN** delicacy
- P3. IF** refrigerated **OR** produce **THEN** perishable
- P4. IF** weighs-15-lbs **AND** inexpensive **AND NOT** perishable **THEN** staple
- P5. IF** perishable **AND** weighs-15-lbs **THEN** turkey
- P6. IF** weighs-15-lbs **AND** produce **THEN** watermelon

INTERPRETER

- Find all productions whose condition parts are true
- Deactivate productions that would add a duplicate symbol
- Execute the lowest numbered production (or quit)
- Repeat

Adapted from the Handbook of AI, Vol 1, pp. 191

Cycle 4

WORKING MEMORY

WM = (turkey, perishable, produce, green, weighs-15-lbs)

CYCLE 4

- Productions whose condition parts are true: **P1, P3, P5, P6**
- Productions **P1, P3, P5** would add duplicate symbol, so deactivate them
- Execute **P6**. This gives: WM = (**watermelon**, turkey, perishable, produce, green, weighs-15-lbs)

RULE MEMORY

- P1. IF** green **THEN** produce
- P2. IF** packed-in-small-container **THEN** delicacy
- P3. IF** refrigerated **OR** produce **THEN** perishable
- P4. IF** weighs-15-lbs **AND** inexpensive **AND NOT** perishable **THEN** staple
- P5. IF** perishable **AND** weighs-15-lbs **THEN** turkey
- P6. IF** weighs-15-lbs **AND** produce **THEN** watermelon

INTERPRETER

- Find all productions whose condition parts are true
- Deactivate productions that would add a duplicate symbol
- Execute the lowest numbered production (or quit)
- Repeat

Adapted from the Handbook of AI, Vol 1, pp. 191

Cycle 5

WORKING MEMORY

WM = (watermelon, turkey, perishable, produce, green, weighs-15-lbs)

CYCLE 5

- Productions whose condition parts are true: **P1, P3, P5, P6**
- Productions **P1, P3, P5, P6** would add duplicate symbol, so **deactivate them**
- Quit.**

RULE MEMORY

- P1. IF** green **THEN** produce
- P2. IF** packed-in-small-container **THEN** delicacy
- P3. IF** refrigerated **OR** produce **THEN** perishable
- P4. IF** weighs-15-lbs **AND** inexpensive **AND NOT** perishable **THEN** staple
- P5. IF** perishable **AND** weighs-15-lbs **THEN** turkey
- P6. IF** weighs-15-lbs **AND** produce **THEN** watermelon

INTERPRETER

- Find all productions whose condition parts are true
- Deactivate productions that would add a duplicate symbol
- Execute the lowest numbered production (or quit)
- Repeat

Adapted from the Handbook of AI, Vol 1, pp. 191

WM = (produce, green, weighs-15-lbs)

CYCLE 2

- Activate: P1, P3, P6
- Deactivate P1
- Execute **P3**. WM = (**perishable**, produce, green, weighs-15-lbs)

CYCLE 3

- Activate: P1, P3, P5, P6
- Deactivate: P1 and P3
- Execute **P5**. WM = (**turkey**, perishable, produce, green, weighs-15-lbs)

Is this a bug in the rules?

CYCLE 4

- Activate: P1, P3, P5, P6
- Deactivate: P1, P3, P5
- Execute **P6**. WM = (**watermelon**, turkey, perishable, produce, green, weighs-15-lbs)

CYCLE 5

- Activate: P1, P3, P5, P6
- Deactivate: P1, P3, P5, P6
- Quit.**

Cycles 2-5

RULE MEMORY

- P1. IF** green **THEN** produce
- P2. IF** packed-in-small-container **THEN** delicacy
- P3. IF** refrigerated **OR** produce **THEN** perishable
- P4. IF** weighs-15-lbs **AND** inexpensive **AND NOT** perishable **THEN** staple
- P5. IF** perishable **AND** weighs-15-lbs **THEN** turkey
- P6. IF** weighs-15-lbs **AND** produce **THEN** watermelon

How the ACT-R production system is more complex

- Watermelon is simple example:
 - Working memory elements*: a single word
 - Production rules*: no variables in if-part
 - Interpreter*: conflict resolution selects lowest numbered unused production
- In contrast, in ACT-R:
 - Working memory elements*: database-like record structures with attributes and values
 - Production rules*: includes variables & patterns
 - Interpreter*: match must deal with variables and patterns, conflict resolution does *not* use rule order

CognitiveTutor/AuthoringTools LearnLab
Pittsburgh School of Learning Center

A second production rule model example

- Think about how you would write production rules to do multi-column addition?

$$\begin{array}{r} 264 \\ + 716 \\ \hline \end{array}$$

- What if-then rules would you write to perform this task in a step-by-step fashion?

1st Annual PSLC Summer School Building a cognitive model in Jess - 13 Pittsburgh, June 27 - July 1, 2005

CognitiveTutor/AuthoringTools LearnLab
Pittsburgh School of Learning Center

Production rules set new goals and perform actions

$$\begin{array}{r} 264 \\ + 716 \\ \hline \end{array}$$

Adapted from Anderson, J. R. 1993. *Rules of the Mind*. Hillsdale, NJ: LEA.

1st Annual PSLC Summer School Building a cognitive model in Jess - 14 Pittsburgh, June 27 - July 1, 2005

Production rule model for addition

FOCUS-ON-FIRST-COLUMN
IF The goal is to do an addition problem
 And there is no pending subgoal
 And there is no result yet in the rightmost column of the problem
THEN Set a subgoal to process the rightmost column

FOCUS-ON-NEXT-COLUMN
IF The goal is to do an addition problem
 And there is no pending subgoal
 And **C** is the rightmost column with numbers to add and no result
THEN Set a subgoal to process column **C**

ADD-ADDENDS
IF There is a goal to process column **C**
THEN Set **Sum** to the sum of the addends in column **C**
 And set a subgoal to write **Sum** as the result in column **C**
 And remove the goal to process column **C**

ADD-CARRY
IF There is a goal to write **Sum** as the result in column **C**
 And there is a carry into column **C**
 And the carry has not been added to **Sum**
THEN Change the goal to write **Sum** + 1 as the result
 And mark the carry as added

MUST-CARRY
IF There is a goal to write **Sum** as the result in column **C**
 And the carry into column **C** (if any) has been added to **Sum**
 And **Sum** > 9
 And **Next** is the column to the left of **C**
THEN Change the goal to write **Sum**-10 as the result in **C**
 Set a subgoal to write 1 as a carry in column **Next**

WRITE-SUM
IF There is a goal to write **Sum** as the result in column **C**
 And **Sum** < 10
 And the carry into column **C** (if any) has been added
THEN Write **Sum** as the result in column **C**
 And remove the goal

WRITE-CARRY
IF There is a goal to write a carry in column **C**
THEN Write the carry in column **C**
 And remove the goal

DONE
IF The goal is to do an addition problem
 And there is no incomplete subgoal to work on
 And there is no column left with numbers to add (or a carry) and no result
THEN Mark the problem as done

1st Annual PSLC Summer School Building a cognitive model in Jess - 15 Pittsburgh, June 27 - July 1, 2005

A Trace of Production Rule Firings

$$\begin{array}{r} \text{column 3} \\ \text{column 2} \\ \text{column 1} \\ 264 \\ + 716 \\ \hline 0 \end{array}$$

Subgoals: Write carry in column2

Step 1
 1. FOCUS-ON-FIRST-COLUMN
 → Goal: Process column1
 2. ADD-ADDENDS
 C = column1
 Sum = 10
 → Goal: Process column1
 → Goal: Write 10 as result in column1
 3. MUST-CARRY
 C = column1
 Next = column2
 → Goal: Write carry in column2
 → Goal: Write 0 as result in column1
 4. WRITE-SUM
 C = column1
 Sum = 0
 Action: Write 0 as result in column1
 → Goal: Write 0 as result in column1

Step 2
 5. WRITE-CARRY
 C = column2
 Action: Write carry in column2
 → Goal: Write carry in column2

Q: Could the carry have been written first?
A: Yes, the condition of WRITE-CARRY holds after step 1.3. The model is flexible w.r.t. the order of writing the carry and writing the result.

Q: Could we have moved on without writing the carry?
A: No, FOCUS-ON-NEXT-COLUMN can fire only if there is no pending goal. The model does NOT allow implicit carrying.

1st Annual PSLC Summer School Building a cognitive model in Jess - 16 Pittsburgh, June 27 - July 1, 2005

A Trace of Production Rule Firings (ctnd.)

$$\begin{array}{r} 1 \\ 264 \\ + 716 \\ \hline 980 \end{array}$$

Step 3
 6. FOCUS-ON-NEXT-COLUMN
 C = column2
 → Goal: Process column2
 7. ADD-ADDENDS
 C = column2
 Sum = 7
 → Goal: Process column2
 → Goal: Write 7 as result in column2
 8. ADD-CARRY
 C = column2
 Sum = 7
 → Goal: Write 8 as result in column2
 9. WRITE-SUM
 C = column2
 Sum = 8
 Action: Write 8 as result in column2
 → Goal: Write 8 as result in column2

Q: Good thing that WRITE-SUM did not fire instead after step 3.7. Why didn't it?
A: WRITE-SUM has condition that carry into the column must have been added.

Step 4
 10. FOCUS-ON-NEXT-COLUMN
 C = column3
 → Goal: Process column3
 11. ADD-ADDENDS
 C = column3
 Sum = 9
 → Goal: Process column3
 → Goal: Write 9 as result in column3
 12. WRITE-SUM
 C = column3
 Sum = 9
 Action: Write 9 as result in column3
 → Goal: Write 9 as result in column3

Step 5
 13. DONE

No pending goal
 DONE

1st Annual PSLC Summer School Building a cognitive model in Jess - 17 Pittsburgh, June 27 - July 1, 2005

CognitiveTutor/AuthoringTools LearnLab
Pittsburgh School of Learning Center

How could you model students that don't carry?

- Instead of doing the addition correctly:

$$\begin{array}{r} 1 \\ 264 \\ + 716 \\ \hline 980 \end{array}$$

- Can you model a student who writes:

$$\begin{array}{r} 264 \\ + 716 \\ \hline 970 \end{array}$$

- How can you change the production rule model?

1st Annual PSLC Summer School Building a cognitive model in Jess - 18 Pittsburgh, June 27 - July 1, 2005