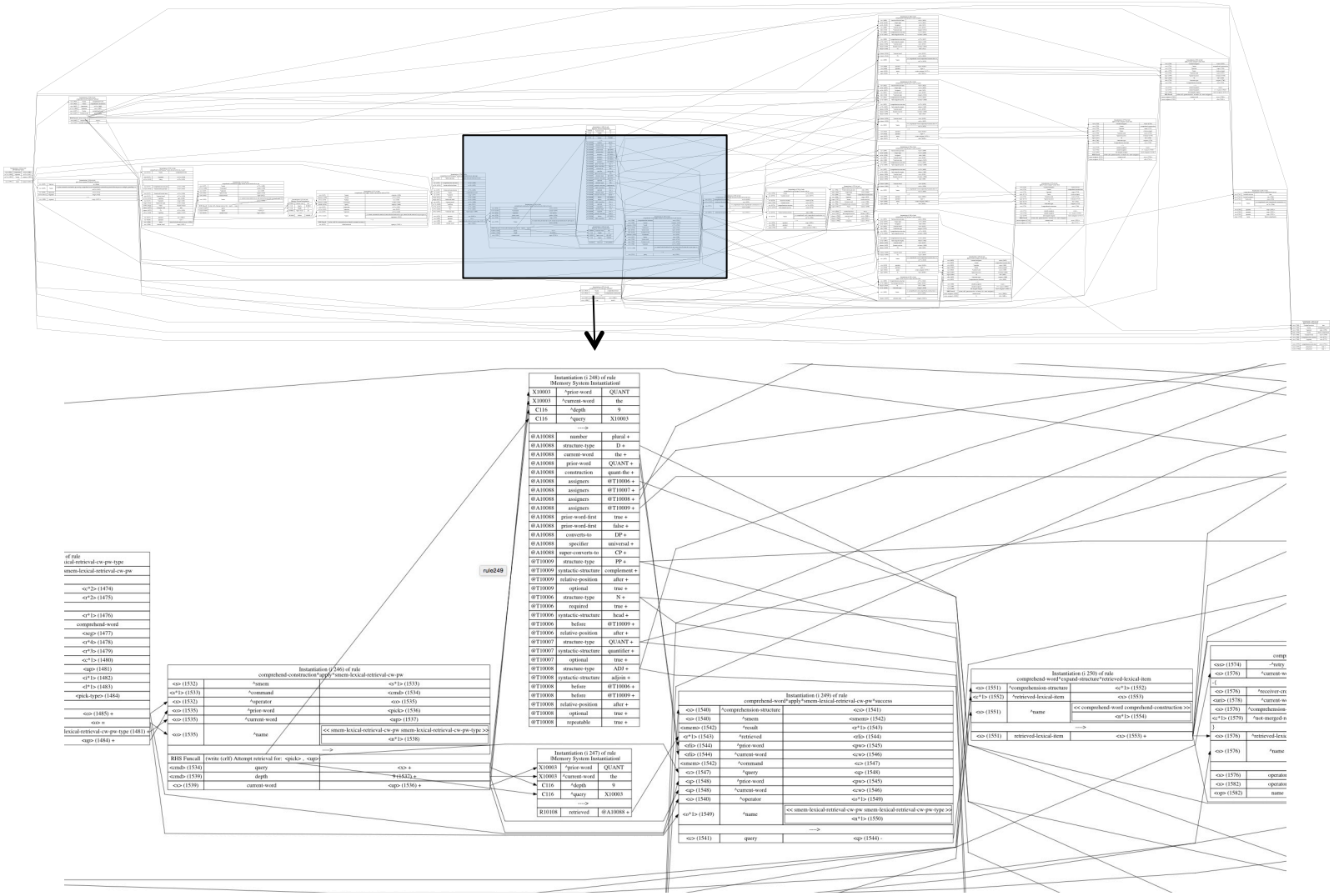
Explaining the Learner

Mazin Assanie
mazina@umich.edu

Understanding How a Rule was Learned is Challenging

- Many different rule firings may be involved.
- Agent engineer must be able to figure out:
  - How those many rules interacted, often transitively across multiple rules
  - Why conditions in particular rules contributed to a learned rule
  - Which parts of WM are accessible to the superstate

Real Instantiation Graphs



Previous Commands

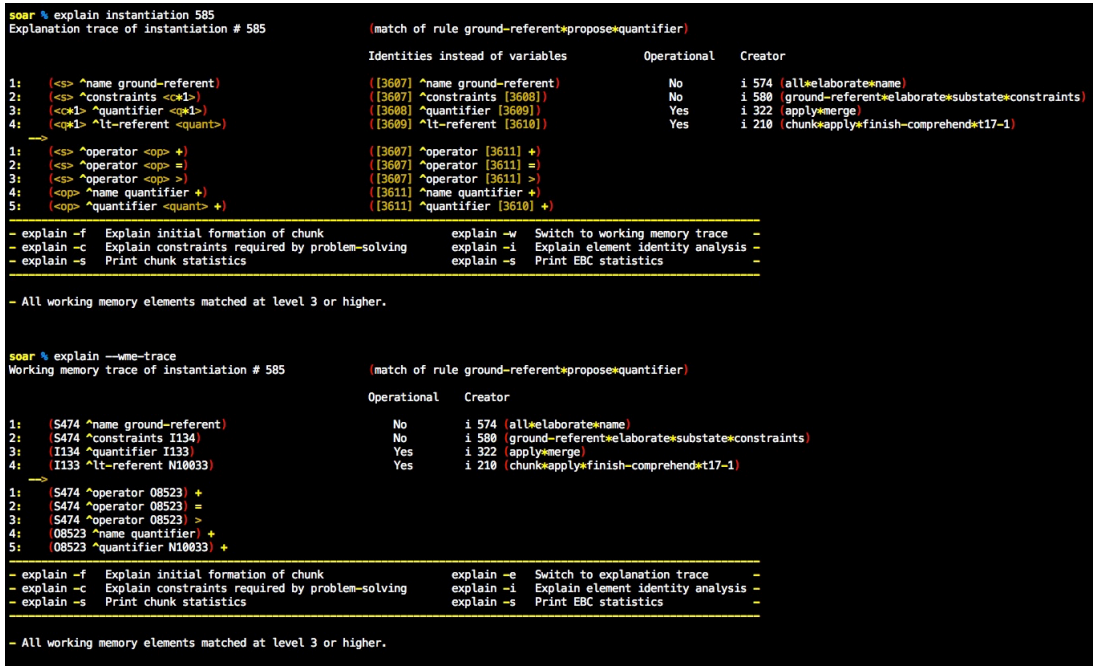
- Trace approach (watch –l)
  - Just conditions being moved around
  - EBC has too much to present for a raw trace
- Log approach (save-backtraces)
  - Was expensive and provided limited information

New Explain Command

- Creates a snapshot of the both the agent's problem-solving when the rule was learned and EBC's analysis of it
  - Two versions of this snapshot: the working memory trace and the explanation trace.

New Explain Command

- Agent engineer can interactively explore this snapshot by incrementally browsing the rule firings that were involved



New Explain Command

- Explainer also provides:
  - High level explanation of chunk formation
  - Many different stats
    - Both global and rule-specific stats
  - EBC-related analysis
    - Identity analysis
    - Constraints required by problem-solving
  - Visualization

How to Use in 3 Easy Steps

1. Tell the Explainer what to pay attention to
   - You can tell it to watch a particular rule or all rules.
   - These should be rules that create results in the superstate

2. Tell the Explainer you want to discuss a rule that it was watching

3. Browse

How to Tell the Explainer to Pay Attention

1. To record all rules learned (includes justifications)

   explain [**--all** | --only-specific]

2. To record a rule at run-time

   explain --record <rule-name>

3. To record a rule from a source file

```
sp {make-result
:explain
   (state <substate> ^superstate <superstate>)
   (<superstate>     ^foo <bar>)
-->
   (<superstate> ^result <bar>)}
```

How to Check What the Explainer Knows

- explain
  - With no arguments, it will print:
    - Summary of explain settings
    - List of last 10 chunks that Soar recorded explanations
- explain --record
  - Print all specific chunks being watched
- explain --list
  - Print the full list of chunks that Soar can explain
  - Can be very long, especially if user is recording justifications
- explain --global

Global Chunking Statistics

```
soar % exp -g
-------------------------
EBC Executions Statistics
-------------------------
Chunks attempted                                21
Chunks successfully built                       18
Chunk failures reverted to justifications       3
Justifications attempted                        44
Justifications successfully built               47

Instantiations built                            2360
Instantiations backtraced through               2183
Instantiations backtraced through twice         11317

Conditions merged                               90
Constraints collected                           735
Constraints attached                            22
Grounding conditions generated                  0

----------------------
EBC Failure Statistics
----------------------
Duplicate chunk already existed                 0
Chunk tested local negation                     63
Could not re-order chunk                        3
Chunk had no grounds                            0
Already reached max-chunks                      0
Chunk formed did not match WM                   0
Justification formed did not match WM           0

-----------------------
EBC Explainer Statistics
-----------------------
Chunks records                                  65
Actions records                                 4358
Condition records                               7951
Instantiation records                           799
```

How to Browse

- In general, you specify what you want to browse using IDs that the Explainer assigns to the Soar data structures it recorded.
  - All explainer output is annotated with these IDs

How to Browse

- You also specify whether you want to browse the working memory trace or the explanation trace
  - **explain [--explanation-trace | --wme-trace]**
- Typically, you'll switch back and forth.
  - Each explainer command prints different information based on the trace involved.

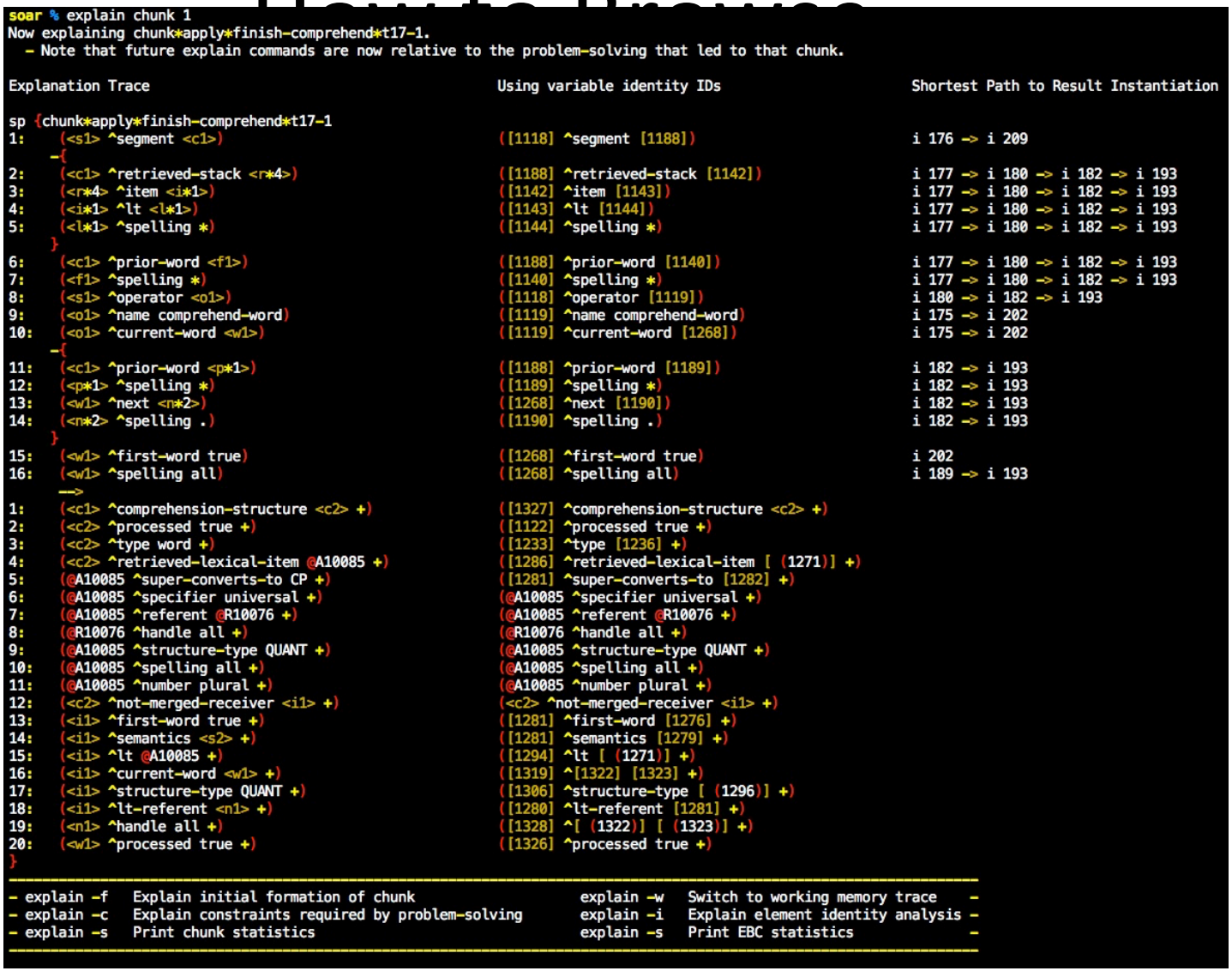
How to Browse

- To start discussing a chunk explanation:
  - explain [chunk name]
  - explain chunk [chunk id number]
    - for e.g., ‘explain c 1’
  - Everything the explainer presents will now be relative to that discussed chunk

How to Browse

```
soar % explain
Explainer Settings:
   Watch all chunk formations                    Yes
   Number of specific rules watched              0

   Chunks available for discussion:                                          justify*check-failure*apply*check-expectation*l2-id
                                                 justify*check-failure*apply*check-expectation*l3-attribute-missing*onc*t280-8 (c53)
                                                 justify*check-failure*apply*check-expectation*l3-attribute-missing*onc*t280-7 (c52)
                                                 justify*check-failure*apply*check-expectation*l2-constant-fail*onc*t280-11 (c56)
                                                 justify*check-failure*apply*check-expectation*l3-attribute-missing*onc*t280-6 (c51)
                                                 justify*check-failure*apply*check-expectation*l3-attribute-missing*onc*t280-5 (c50)
                                                 justify*apply*check-expectation*new-message*onc*t280-18 (c63)
                                                 justify*check-failure*apply*check-expectation*l2-attribute-missing*onc*t280-10 (c55)
                                                 justify*check-failure*apply*check-expectation*l3-identifier-fail*onc*t280-3 (c48)
                                                 justify*check-failure*apply*check-expectation*l2-attribute-missing*onc*t280-2 (c47)

* Note:  Only printed the first 10 chunks.  Type 'explain --list' to see the other 55 chunks.

   Current chunk being discussed:                chunk*apply*finish-comprehend*t31-1(c2)

Type 'explain [chunk-name]' or 'explain c [chunk id]' to discuss the formation of that chunk.
```

How to Prevent



```
soar % explain chunk 1
Now explaining chunk*apply*finish-comprehend*t17-1.
  - Note that future explain commands are now relative to the problem-solving that led to that chunk.

Explanation Trace                                    Using variable identity IDs                    Shortest Path to Result Instantiation

sp {chunk*apply*finish-comprehend*t17-1
1:    (<s1> ^segment <c1>)                           ([1118] ^segment [1188])                       i 176 -> i 209
     -{
2:    (<c1> ^retrieved-stack <r*4>)                  ([1188] ^retrieved-stack [1142])               i 177 -> i 180 -> i 182 -> i 193
3:    (<r*4> ^item <i*1>)                            ([1142] ^item [1143])                          i 177 -> i 180 -> i 182 -> i 193
4:    (<i*1> ^lt <l*1>)                              ([1143] ^lt [1144])                            i 177 -> i 180 -> i 182 -> i 193
5:    (<l*1> ^spelling *)                            ([1144] ^spelling *)                           i 177 -> i 180 -> i 182 -> i 193
     }
6:    (<c1> ^prior-word <f1>)                        ([1188] ^prior-word [1140])                    i 177 -> i 180 -> i 182 -> i 193
7:    (<f1> ^spelling *)                             ([1140] ^spelling *)                           i 177 -> i 180 -> i 182 -> i 193
8:    (<s1> ^operator <o1>)                          ([1118] ^operator [1119])                      i 180 -> i 182 -> i 193
9:    (<o1> ^name comprehend-word)                   ([1119] ^name comprehend-word)                 i 175 -> i 202
10:   (<o1> ^current-word <w1>)                      ([1119] ^current-word [1268])                  i 175 -> i 202
     -{
11:   (<c1> ^prior-word <p*1>)                       ([1188] ^prior-word [1189])                    i 182 -> i 193
12:   (<p*1> ^spelling *)                           ([1189] ^spelling *)                           i 182 -> i 193
13:   (<w1> ^next <n*2>)                            ([1268] ^next [1190])                          i 182 -> i 193
14:   (<n*2> ^spelling .)                           ([1190] ^spelling .)                           i 182 -> i 193
     }
15:   (<w1> ^first-word true)                       ([1268] ^first-word true)                      i 202
16:   (<w1> ^spelling all)                          ([1268] ^spelling all)                         i 189 -> i 193
      -->
1:    (<c1> ^comprehension-structure <c2> +)         ([1327] ^comprehension-structure <c2> +)
2:    (<c2> ^processed true +)                       ([1122] ^processed true +)
3:    (<c2> ^type word +)                            ([1233] ^type [1236] +)
4:    (<c2> ^retrieved-lexical-item @A10085 +)       ([1286] ^retrieved-lexical-item [ (1271)] +)
5:    (@A10085 ^super-converts-to CP +)              ([1281] ^super-converts-to [1282] +)
6:    (@A10085 ^specifier universal +)               (@A10085 ^specifier universal +)
7:    (@A10085 ^referent @R10076 +)                  (@A10085 ^referent @R10076 +)
8:    (@R10076 ^handle all +)                        (@R10076 ^handle all +)
9:    (@A10085 ^structure-type QUANT +)              (@A10085 ^structure-type QUANT +)
10:   (@A10085 ^spelling all +)                      (@A10085 ^spelling all +)
11:   (@A10085 ^number plural +)                     (@A10085 ^number plural +)
12:   (<c2> ^not-merged-receiver <i1> +)            (<c2> ^not-merged-receiver <i1> +)
13:   (<i1> ^first-word true +)                     ([1281] ^first-word [1276] +)
14:   (<i1> ^semantics <s2> +)                      ([1281] ^semantics [1279] +)
15:   (<i1> ^lt @A10085 +)                          ([1294] ^lt [ (1271)] +)
16:   (<i1> ^current-word <w1> +)                   ([1319] ^[1322] [1323] +)
17:   (<i1> ^structure-type QUANT +)                ([1306] ^structure-type [ (1296)] +)
18:   (<i1> ^lt-referent <n1> +)                    ([1280] ^lt-referent [1281] +)
19:   (<n1> ^handle all +)                         ([1328] ^[ (1322)] [ (1323)] +)
20:   (<w1> ^processed true +)                     ([1326] ^processed true +)
}

-------------------------------------------------------------------------------------------------
- explain -f   Explain initial formation of chunk                 explain -w   Switch to working memory trace    -
- explain -c   Explain constraints required by problem-solving    explain -i   Explain element identity analysis -
- explain -s   Print chunk statistics                             explain -s   Print EBC statistics             -
-------------------------------------------------------------------------------------------------
```

How to Browse

- To examine EBC's high-level analysis of the problem-solving underlying the learned rule:
  - explain --formation
    - Most likely starting point
  - explain --identity
  - explain --stats
  - explain --constraints
    - Still in progress

How to Browse

```
soar % explain -f
The formation of 'chunk*ground-referent*apply*failed-grounding*finish3*onc*t75-5' (c14):

   (1) At time 75, rule 'ground-referent*apply*failed-grounding*finish3' matched (i 601)
       and created results in a superstate.

   (2) Conditions of base instantiations and any relevant operator selection knowledge
       are backtraced through to determine what superstate knowledge was tested and
       should appear in the chunk.

   (3) EBC analyzes the how variables are used in the explanation trace to determine
       sets of elements that share the same semantics.  (explain -i)

   (4) EBC constraint analysis is performed to determine all constraints on the
       values of variables in (3) that were required by problem-solving (explain -c).

   (5) The mappings determined by the identity set analysis are used to variablize
       conditions collected in (2) and constraints collected in (4).  (explain -i)

   (6) Generalized constraints from (5) are added and final chunk conditions are
       polished by pruning unnecessary tests and merging appropriate conditions.
       Statistics on polishing are available.  (explain -c) (explain -s)

------------------------------------------------------------------------------

The following 1 instantiations fired to produce results in Step (2)

Initial base instantiation i 601 that fired when ground-referent*apply*failed-grounding*finish3 matched at time 75:

Explanation trace of instantiation # 601              (match of rule ground-referent*apply*failed-grounding*finish3)

                                                      Identities instead of variables           Operational    Creator

1:    (<s> ^proto-referent <ref>)                     ([3686] ^proto-referent [3687])               No         i 589 (ground-referent*apply*quantifier)
2:    (<ref> ^predicates-copied true)                 ([3687] ^predicates-copied true)              Yes        Higher-level Problem Space
3:    (<s> ^segment <ss>)                             ([3686] ^segment [3688])                      No         i 582 (pass-down-segment*elaborate*substate)
4:    (<ss> ^dialog-object-list-access <dol>)         ([3688] ^dialog-object-list-access [3689])    Yes        Higher-level Problem Space
5:    (<s> ^operator <o*1>)                           ([3686] ^operator [3690])                     No         i 585 (ground-referent*propose*quantifier)
6:    (<o*1> ^name { << create-hypothetical quantifier failed-grounding >> <n*1> })                 ([3690] ^name { << create-hypothetical quantifier failed-gr
-referent*propose*quantifier)
   -->
1:    (<ss> ^dialog-object-list-access <dol> -)       ([3688] ^dialog-object-list-access [3689] -)
2:    (<ss> ^dialog-object-list-access <ndol> +)      ([3688] ^dialog-object-list-access [3692] +)
3:    (<ndol> ^referent <ref> +)                      ([3692] ^referent [3687] +)
4:    (<ndol> ^next <dol> +)                          ([3692] ^next [3689] +)

All 8 rule firings involved in problem solving:

   i 580 (ground-referent*elaborate*substate*constraints)
   i 589 (ground-referent*apply*quantifier)
   i 582 (pass-down-segment*elaborate*substate)
   i 585 (ground-referent*propose*quantifier)
   i 571 (elaborate*state*top-state)
   i 574 (all*elaborate*name)
   i 598 (ground-referent*apply*failed-grounding*property)
   i 601 (ground-referent*apply*failed-grounding*finish3)
```

How to Browse

```
soar % e -s
Statistics for 'chunk*ground-referent*apply*failed-grounding*finish3*onc*t75-5' (c14):

Number of conditions                        15
Number of actions                           4
Base instantiation                          i 601 (ground-referent*apply*failed-grounding*finish3)
Extra result instantiations
Instantiations backtraced through                       8
Instantiations backtraced multiple times    13
Conditions merged                           3
Constraints collected                       6
Constraints attached                        2
Grounding conditions added                  0

Duplicates chunks later created             0
Tested negation in local substate           Yes
Failed chunk reverted to justification      No
```

How to Browse

- Two most common command you'll use
  - Print an instantiation
    - explain instantiation 3
      - explain i 3
  - Switch between the explanation and working memory trace
    - explain [--explanation-trace | --wme-trace]

How to Browse

```
soar % explain instantiation 585
Explanation trace of instantiation # 585              (match of rule ground-referent*propose*quantifier)

                                                      Identities instead of variables          Operational    Creator

1:    (<s> ^name ground-referent)                     ([3607] ^name ground-referent)               No         i 574 (all*elaborate*name)
2:    (<s> ^constraints <c*1>)                        ([3607] ^constraints [3608])                No         i 580 (ground-referent*elaborate*substate*constraints)
3:    (<c*1> ^quantifier <q*1>)                       ([3608] ^quantifier [3609])                 Yes        i 322 (apply*merge)
4:    (<q*1> ^lt-referent <quant>)                    ([3609] ^lt-referent [3610])                Yes        i 210 (chunk*apply*finish-comprehend*t17-1)
  -->
1:    (<s> ^operator <op> +)                          ([3607] ^operator [3611] +)
2:    (<s> ^operator <op> =)                          ([3607] ^operator [3611] =)
3:    (<s> ^operator <op> >)                          ([3607] ^operator [3611] >)
4:    (<op> ^name quantifier +)                       ([3611] ^name quantifier +)
5:    (<op> ^quantifier <quant> +)                    ([3611] ^quantifier [3610] +)
-------------------------------------------------------------------------------------------------------------
- explain -f   Explain initial formation of chunk                 explain -w   Switch to working memory trace    -
- explain -c   Explain constraints required by problem-solving    explain -i   Explain element identity analysis -
- explain -s   Print chunk statistics                             explain -s   Print EBC statistics             -
-------------------------------------------------------------------------------------------------------------

- All working memory elements matched at level 3 or higher.



soar % explain --wme-trace
Working memory trace of instantiation # 585           (match of rule ground-referent*propose*quantifier)

                                                      Operational    Creator

1:    (S474 ^name ground-referent)                        No         i 574 (all*elaborate*name)
2:    (S474 ^constraints I134)                            No         i 580 (ground-referent*elaborate*substate*constraints)
3:    (I134 ^quantifier I133)                             Yes        i 322 (apply*merge)
4:    (I133 ^lt-referent N10033)                          Yes        i 210 (chunk*apply*finish-comprehend*t17-1)
  -->
1:    (S474 ^operator O8523) +
2:    (S474 ^operator O8523) =
3:    (S474 ^operator O8523) >
4:    (O8523 ^name quantifier) +
5:    (O8523 ^quantifier N10033) +
-------------------------------------------------------------------------------------------------------------
- explain -f   Explain initial formation of chunk                 explain -e   Switch to explanation trace       -
- explain -c   Explain constraints required by problem-solving    explain -i   Explain element identity analysis -
- explain -s   Print chunk statistics                             explain -s   Print EBC statistics             -
-------------------------------------------------------------------------------------------------------------

- All working memory elements matched at level 3 or higher.
```

Efficiency

- Much effort was put to minimize computation and memory costs
- Shared instantiations
  - Different chunk explanations share the same instantiation snapshots if they were created in the same substate.
- Lazy recording of productions
  - Does not record productions unless necessary, extracting production from RETE when available
  - If a rule is excised from the RETE and some explanation depends on it, the Explainer will cache a copy first.

Efficiency

- Delayed recording
  - Records very little information until EBC knows a chunk is formed and added to the rete.
- Cap on the number of instantiations that can be recorded per chunk
- Shallow copying of certain data structures

Nuggets

- Was very effective for tracking down very convoluted chunking issues that arose in the complex Rosie agents
- Even if we record every single chunk and justification formed for one of our complex agents, we don't notice an obvious slow-down.

Nuggets

- Very useful for debugging
- Was very tricky to get working. The explainer code is probably as complex as the actual EBC code. And we did it fairly quickly.
- It set up the basis for visualizing chunk formation (next talk)

Coal

- You still need to understand how EBC works fairly well
- Identity analysis presentation needs work
- Constraint analysis presentation needs implementation
- Some rules with very long conditions are difficult to print out in a clear way
  - Needs to truncate or more dynamic column widths
- No concrete data on the cost of using this option