Episodic-Memory Tutorial

Soar Workshop 32 – Nate Derbinsky

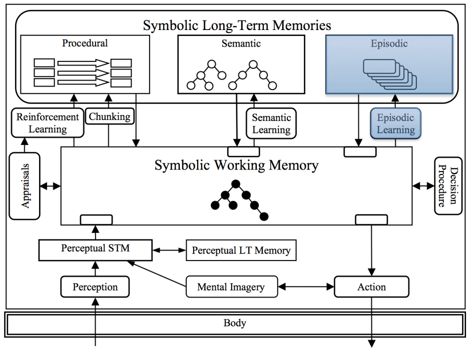
While waiting…

1. Make sure you have internet access
2. Download Soar Tutorial package v9.3.2
   code.google.com/p/soar/wiki/SoarTutorial
3. Download Graphviz
   www.graphviz.org
4. Download Eclipse (with at least Java)
   www.eclipse.org
5. Download tutorial support files
   web.eecs.umich.edu/~nlderbin/workshop32

Agenda

- Big picture
- Basic usage
- Demo task
- Additional resources

Soar 9

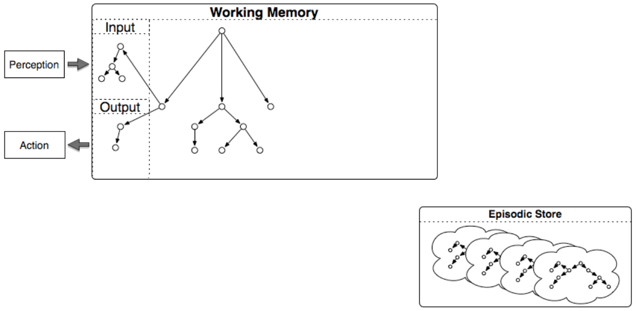


Episodic Memory: Big Picture

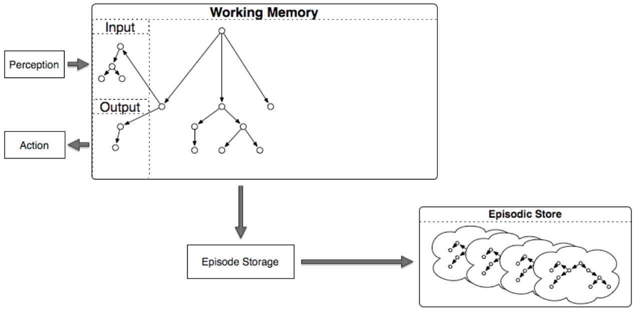
Episodic memory is a weak learning mechanism

- Automatically captures, stores, and temporally indexes agent state
- Supports content-addressable agent interface to autobiographical prior experience

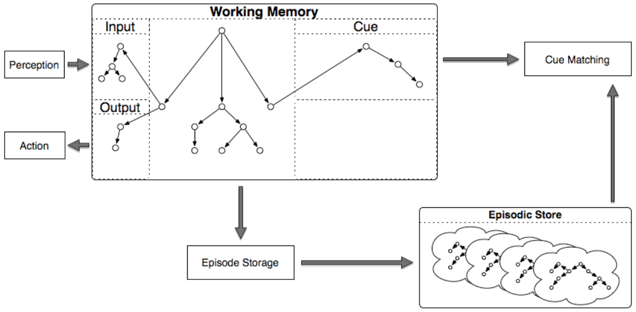
Architectural Integration



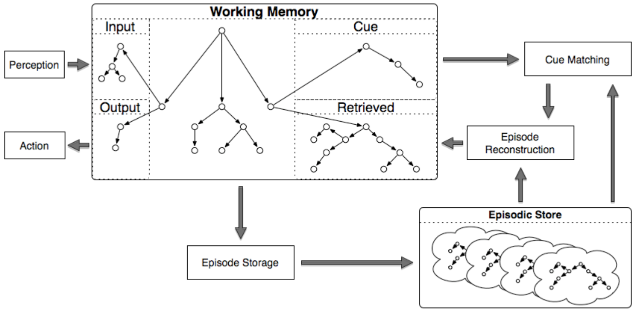
Architectural Integration



Architectural Integration



Architectural Integration



Basic Usage

- Working-memory structure
- Episodic-memory representation
- Controlling episodic memory
- Storing knowledge
- Retrieving knowledge

Working-Memory Structure

Soar creates an `epmem` structure on each state

- Soar Java Debugger
  - `step 5`
  - `print --exact (* ^epmem *)`
  - `print e1`

Each `epmem` structure has specialized substructure

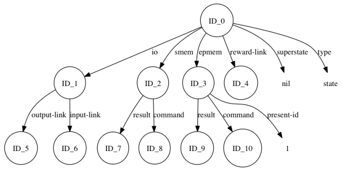
- `command`: agent-initiated actions
- `result`: architectural feedback
- `present-id`: current episode number (more later)

Episodic-Memory Representation

Similar to working memory: symbolic triples

- Attributes cannot be identifiers (currently)
- Structures within an episode are connected; separate episodes are disconnected

```
(<id0> ^epmem <id3>
       ^io <id1>
       ^reward-link <id4>
       ^smem <id2>
       ^superstate nil
       ^type state)
(<id1> ^input-link <id6>
       ^output-link <id5>)
(<id2> ^command <id8> ^result <id7>)
(<id3> ^command <id10> ^present-id 1
       ^result <id9>)
```



Controlling Episodic Memory

Get/Set a parameter:

- `epmem [-g|--get] <name>`
- `epmem [-s|--set] <name> <value>`

EpMem is **disabled** by default. Try enabling it…

1. `epmem`
2. `epmem --set learning on`
3. `epmem`

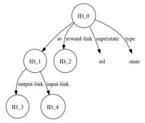
Storing Knowledge

- Automatic storage requires EpMem to be **enabled** (see slide 12)
- Storage captures the top state of working memory
- Events trigger storage of new episodes
  - `epmem --set trigger << dc output >>`
    - `dc`: decision cycle
    - `output`: new augmentation of output-link (default)
- Storage takes place at the end of a phase
  - `epmem --set phase << output selection >>`
    - `output` is default
    - `selection` may be useful for in-the-head agents

Automatic Storage: Example (1)

- Soar Java Debugger

```
1. epmem --set trigger dc
2. epmem --set learning on
3. watch --epmem
4. run 5 -p
5. epmem --print 1
6. ctf ep.gv epmem --viz 1
7. print e1
8. epmem --stats
```



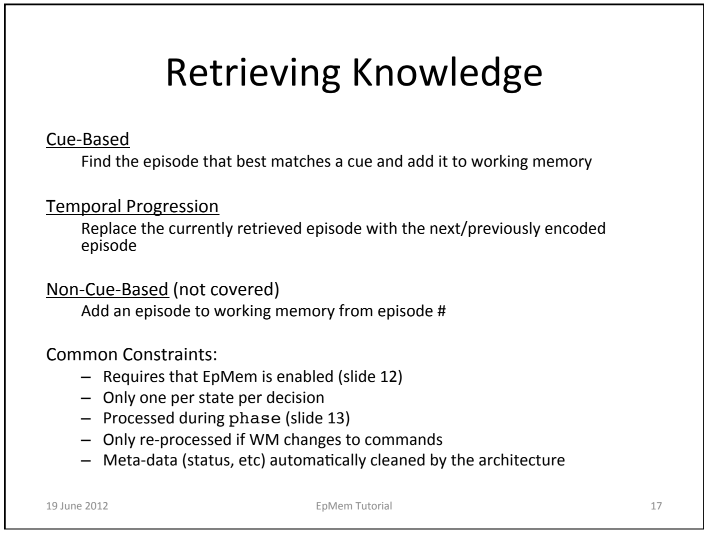
Automatic Storage: Example (1)

Debrief

- What wasn't captured?
- Attributes can be excluded from encoding (and subsequent recursion)
  - `epmem --set exclusions <label>`
    - If `<label>` already excluded, now included
- Try previous example, add:
  - `epmem --set exclusions epmem`
  - `epmem --set exclusions smem`

Automatic Storage: Example (2)

- Eaters
  1. New agent (advanced-move.soar)
     - Spawn Debugger
  2. `epmem --set learning on`
  3. `run`
  4. `epmem --stats`



Retrieving Knowledge

<u>Cue-Based</u>

Find the episode that best matches a cue and add it to working memory

<u>Temporal Progression</u>

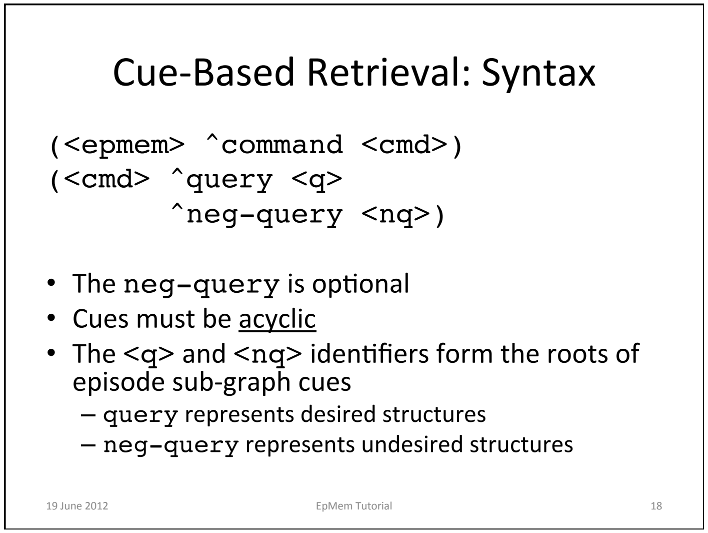
Replace the currently retrieved episode with the next/previously encoded episode

<u>Non-Cue-Based</u> (not covered)

Add an episode to working memory from episode #

Common Constraints:

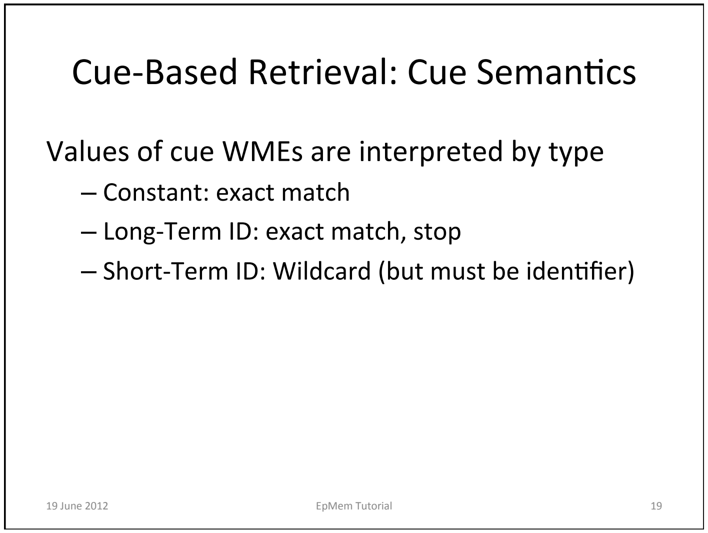
- Requires that EpMem is enabled (slide 12)
- Only one per state per decision
- Processed during `phase` (slide 13)
- Only re-processed if WM changes to commands
- Meta-data (status, etc) automatically cleaned by the architecture



Cue-Based Retrieval: Syntax

```
(<epmem> ^command <cmd>)
(<cmd> ^query <q>
       ^neg-query <nq>)
```

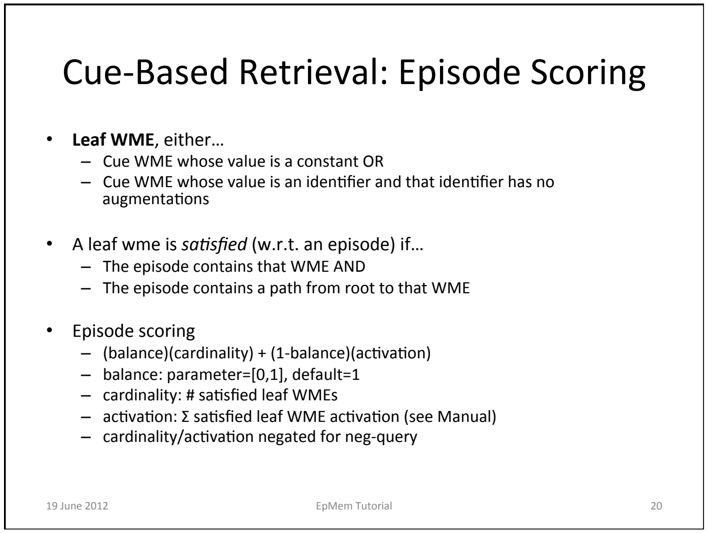
- The `neg-query` is optional
- Cues must be <u>acyclic</u>
- The <`q`> and <`nq`> identifiers form the roots of episode sub-graph cues
  - `query` represents desired structures
  - `neg-query` represents undesired structures



Cue-Based Retrieval: Cue Semantics

Values of cue WMEs are interpreted by type

- Constant: exact match
- Long-Term ID: exact match, stop
- Short-Term ID: Wildcard (but must be identifier)



Cue-Based Retrieval: Episode Scoring

- **Leaf WME**, either…
  - Cue WME whose value is a constant OR
  - Cue WME whose value is an identifier and that identifier has no augmentations
- A leaf wme is *satisfied* (w.r.t. an episode) if…
  - The episode contains that WME AND
  - The episode contains a path from root to that WME
- Episode scoring
  - (balance)(cardinality) + (1-balance)(activation)
  - balance: parameter=[0,1], default=1
  - cardinality: # satisfied leaf WMEs
  - activation: Σ satisfied leaf WME activation (see Manual)
  - cardinality/activation negated for neg-query

Cue-Based Retrieval: Cue Matching

<u>Graph matching</u>

```
epmem --set graph-match << on off >>
```

- on by default

<u>Candidate episode</u>

Defined as satisfying at least one leaf WME

Cue matching will return the most recent graph-matched episode, or the most recent non-graph-matched candidate episode with the maximal episode score

Cue-Based Retrieval: Result

Augmentation	Meaning
^retrieved <retrieval-root>	Root of the retrieved memory
^<< success failure >> <query> <neg-query>	Query status
^match-score #	Float, episode score (slide 19)
^cue-size #	Integer, number of leaf WMEs
^normalized-match-score #	match-score/cue-size
^match-cardinality #	Integer, number of satisfied leaf WMEs (|query| - |neg-query|)
^memory-id #	Integer, episode # retrieved
^present-id #	Integer, current episode #
^graph-match << 0 1 >>	Integer, 1 if graph match succeeded
^mapping <mapping-root>	A mapping from the cue to episode

Cue-Based Retrieval: Example (1)

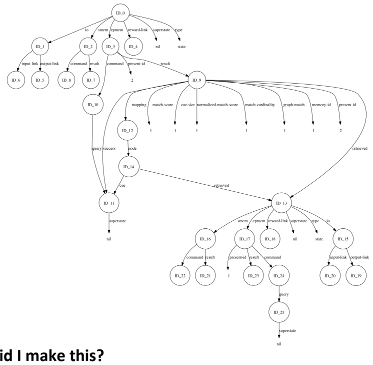
query-1.soar (find superstate nil)

- Soar Java Debugger

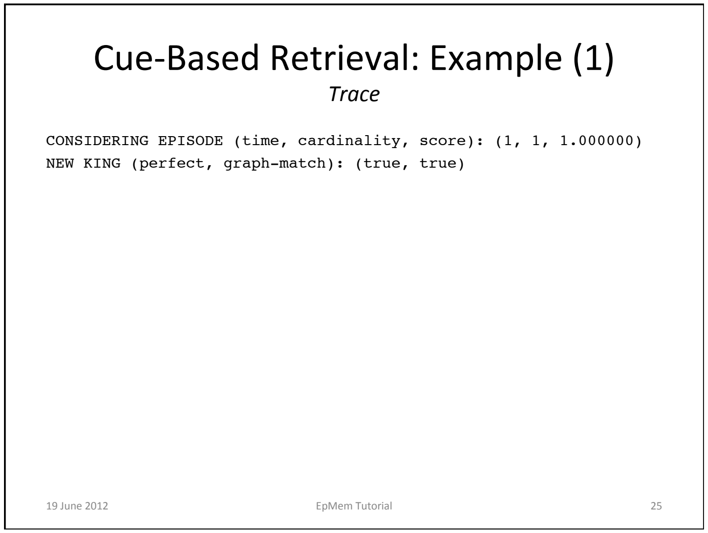
```
1. epmem --set trigger dc
2. epmem --set learning on
3. watch --epmem
4. sp {query1
      (state <s> ^superstate nil
                 ^epmem.command <cmd>)
   -->
      (<cmd> ^query.superstate nil)}
5. run 5 -p
6. print -d 10 e1
```

Cue-Based Retrieval: Example (1)

Result



Pop Quiz: how did I make this?

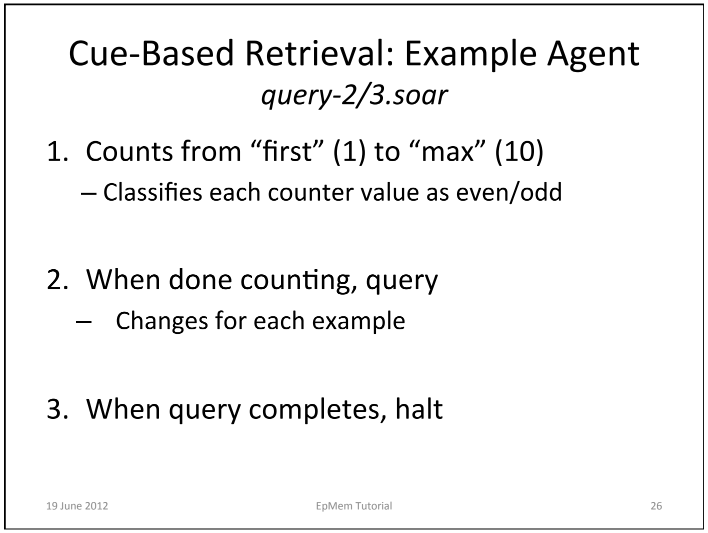


Cue-Based Retrieval: Example (1)

Trace

```
CONSIDERING EPISODE (time, cardinality, score): (1, 1, 1.000000)
NEW KING (perfect, graph-match): (true, true)
```

19 June 2012 EpMem Tutorial 25

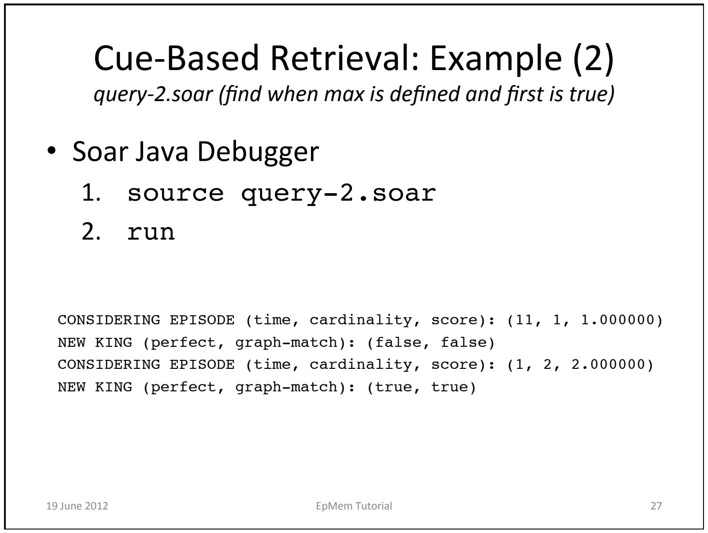


Cue-Based Retrieval: Example Agent

query-2/3.soar

1. Counts from “first” (1) to “max” (10)
   – Classifies each counter value as even/odd
2. When done counting, query
   – Changes for each example
3. When query completes, halt

19 June 2012 EpMem Tutorial 26



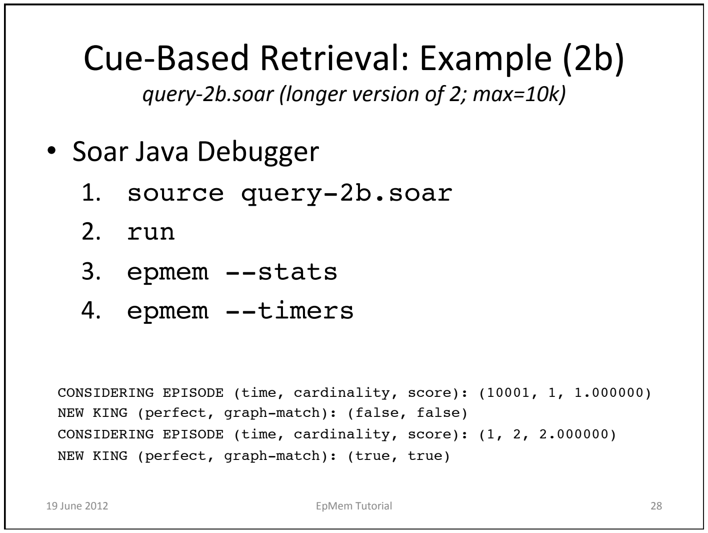
Cue-Based Retrieval: Example (2)

query-2.soar (find when max is defined and first is true)

- Soar Java Debugger
  1. `source query-2.soar`
  2. `run`

```
CONSIDERING EPISODE (time, cardinality, score): (11, 1, 1.000000)
NEW KING (perfect, graph-match): (false, false)
CONSIDERING EPISODE (time, cardinality, score): (1, 2, 2.000000)
NEW KING (perfect, graph-match): (true, true)
```

19 June 2012 EpMem Tutorial 27



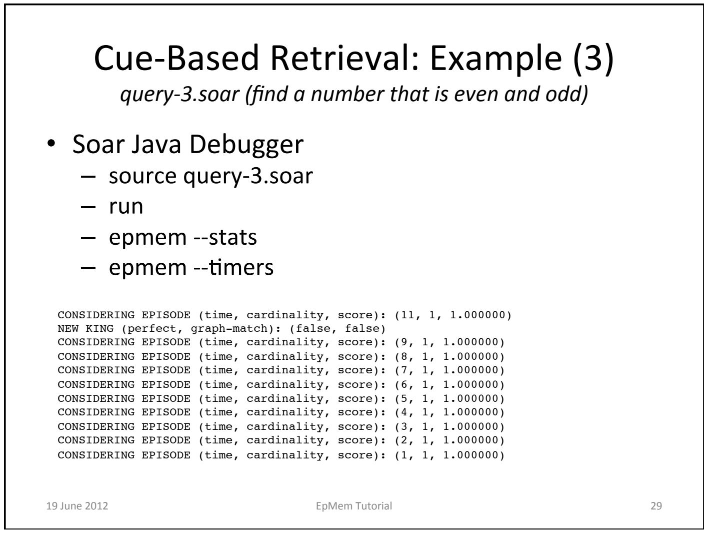
Cue-Based Retrieval: Example (2b)

query-2b.soar (longer version of 2; max=10k)

- Soar Java Debugger
  1. `source query-2b.soar`
  2. `run`
  3. `epmem --stats`
  4. `epmem --timers`

```
CONSIDERING EPISODE (time, cardinality, score): (10001, 1, 1.000000)
NEW KING (perfect, graph-match): (false, false)
CONSIDERING EPISODE (time, cardinality, score): (1, 2, 2.000000)
NEW KING (perfect, graph-match): (true, true)
```

19 June 2012 EpMem Tutorial 28

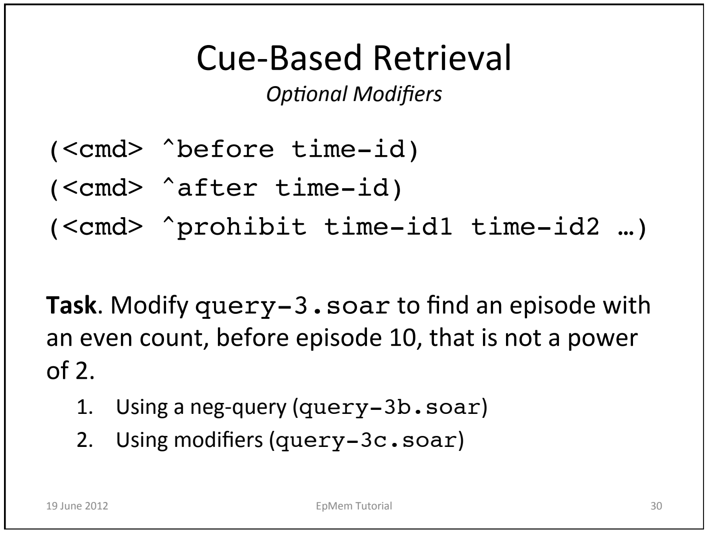


Cue-Based Retrieval: Example (3)

query-3.soar (find a number that is even and odd)

- Soar Java Debugger
  - source query-3.soar
  - run
  - epmem --stats
  - epmem --timers

```
CONSIDERING EPISODE (time, cardinality, score): (11, 1, 1.000000)
NEW KING (perfect, graph-match): (false, false)
CONSIDERING EPISODE (time, cardinality, score): (9, 1, 1.000000)
CONSIDERING EPISODE (time, cardinality, score): (8, 1, 1.000000)
CONSIDERING EPISODE (time, cardinality, score): (7, 1, 1.000000)
CONSIDERING EPISODE (time, cardinality, score): (6, 1, 1.000000)
CONSIDERING EPISODE (time, cardinality, score): (5, 1, 1.000000)
CONSIDERING EPISODE (time, cardinality, score): (4, 1, 1.000000)
CONSIDERING EPISODE (time, cardinality, score): (3, 1, 1.000000)
CONSIDERING EPISODE (time, cardinality, score): (2, 1, 1.000000)
CONSIDERING EPISODE (time, cardinality, score): (1, 1, 1.000000)
```



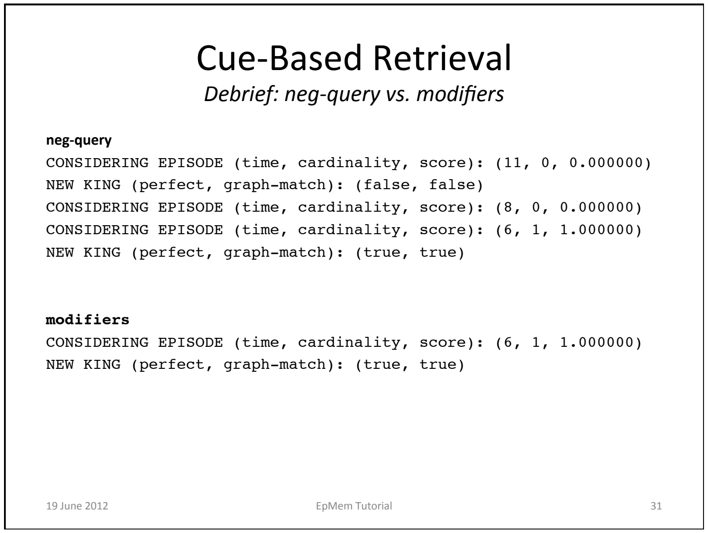
Cue-Based Retrieval

Optional Modifiers

```
(<cmd> ^before time-id)
(<cmd> ^after time-id)
(<cmd> ^prohibit time-id1 time-id2 …)
```

Task. Modify `query-3.soar` to find an episode with an even count, before episode 10, that is not a power of 2.

1. Using a neg-query (`query-3b.soar`)
2. Using modifiers (`query-3c.soar`)



Cue-Based Retrieval

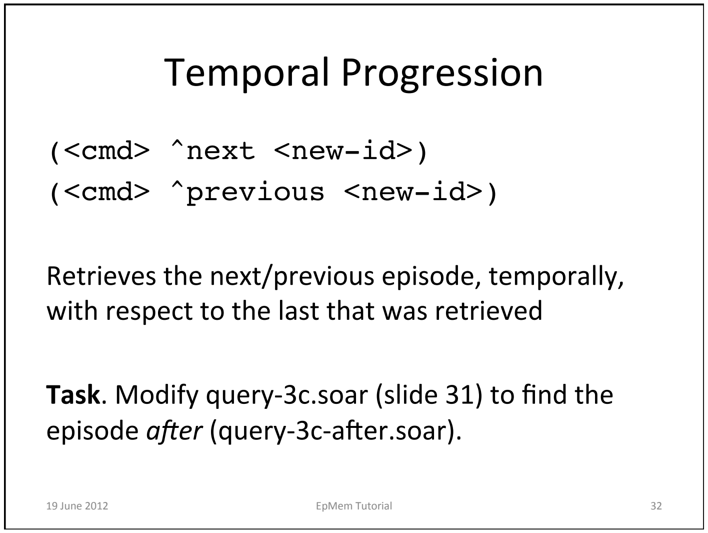
Debrief: neg-query vs. modifiers

neg-query

```
CONSIDERING EPISODE (time, cardinality, score): (11, 0, 0.000000)
NEW KING (perfect, graph-match): (false, false)
CONSIDERING EPISODE (time, cardinality, score): (8, 0, 0.000000)
CONSIDERING EPISODE (time, cardinality, score): (6, 1, 1.000000)
NEW KING (perfect, graph-match): (true, true)
```

modifiers

```
CONSIDERING EPISODE (time, cardinality, score): (6, 1, 1.000000)
NEW KING (perfect, graph-match): (true, true)
```

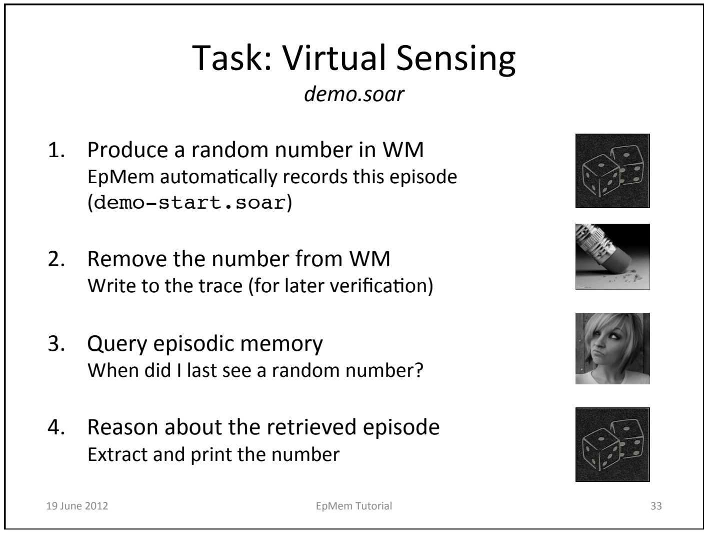


Temporal Progression

```
(<cmd> ^next <new-id>)
(<cmd> ^previous <new-id>)
```

Retrieves the next/previous episode, temporally, with respect to the last that was retrieved

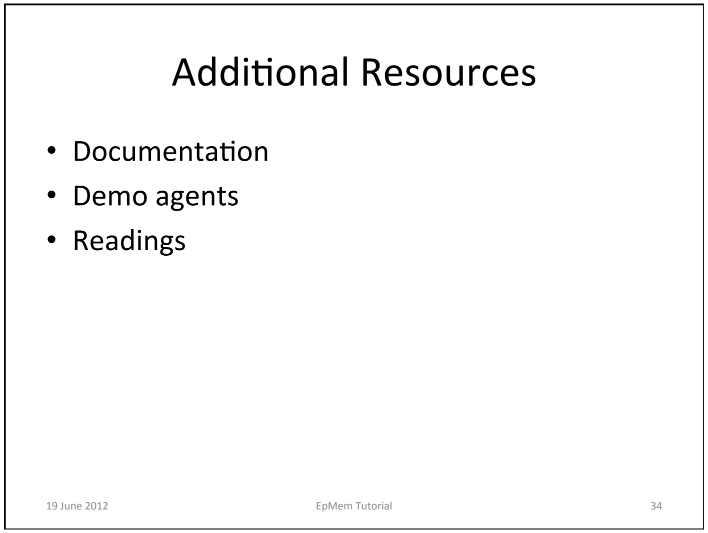
Task. Modify query-3c.soar (slide 31) to find the episode *after* (query-3c-after.soar).



Task: Virtual Sensing

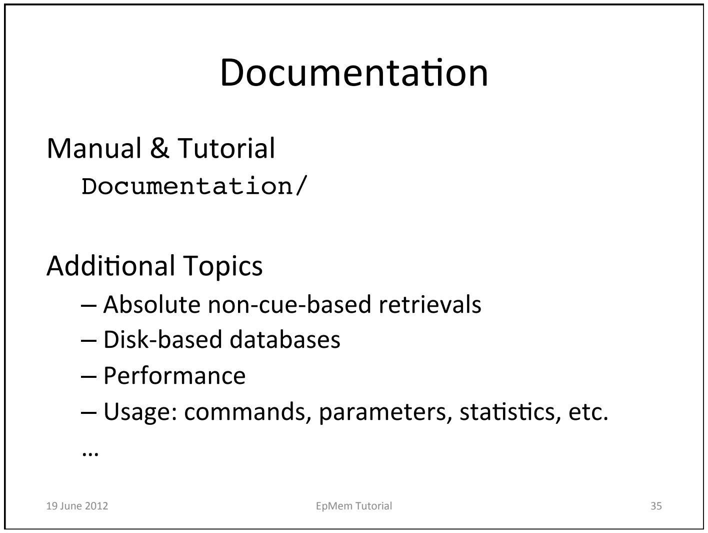
demo.soar

1. Produce a random number in WM
   EpMem automatically records this episode
   (`demo-start.soar`)
2. Remove the number from WM
   Write to the trace (for later verification)
3. Query episodic memory
   When did I last see a random number?
4. Reason about the retrieved episode
   Extract and print the number



Additional Resources

- Documentation
- Demo agents
- Readings



Documentation

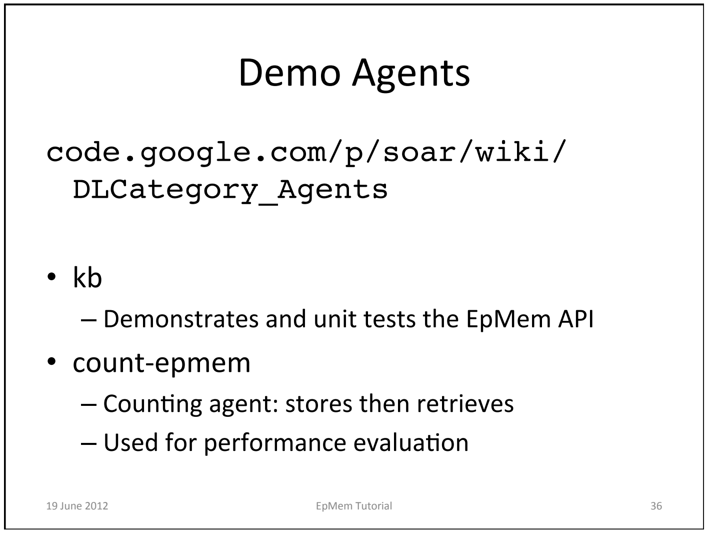
Manual & Tutorial

`Documentation/`

Additional Topics

- Absolute non-cue-based retrievals
- Disk-based databases
- Performance
- Usage: commands, parameters, statistics, etc.

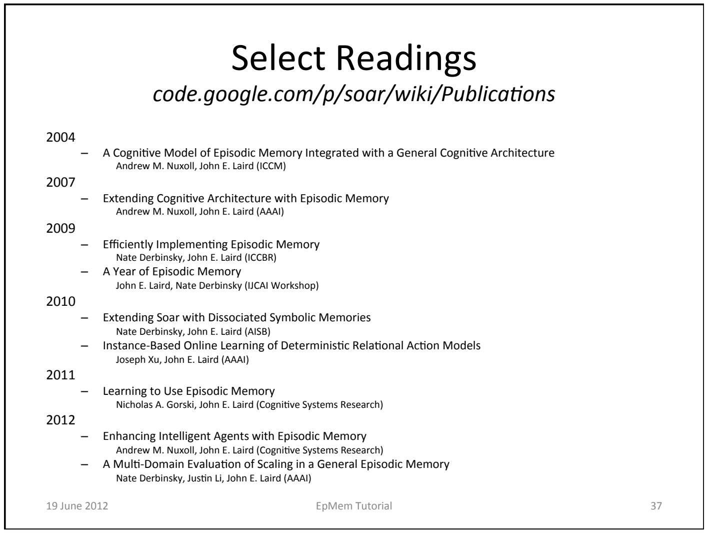
…



Demo Agents

`code.google.com/p/soar/wiki/`
`DLCategory_Agents`

- kb
  - Demonstrates and unit tests the EpMem API
- count-epmem
  - Counting agent: stores then retrieves
  - Used for performance evaluation



Select Readings

code.google.com/p/soar/wiki/Publications

2004

- A Cognitive Model of Episodic Memory Integrated with a General Cognitive Architecture
  Andrew M. Nuxoll, John E. Laird (ICCM)

2007

- Extending Cognitive Architecture with Episodic Memory
  Andrew M. Nuxoll, John E. Laird (AAAI)

2009

- Efficiently Implementing Episodic Memory
  Nate Derbinsky, John E. Laird (ICCBR)
- A Year of Episodic Memory
  John E. Laird, Nate Derbinsky (IJCAI Workshop)

2010

- Extending Soar with Dissociated Symbolic Memories
  Nate Derbinsky, John E. Laird (AISB)
- Instance-Based Online Learning of Deterministic Relational Action Models
  Joseph Xu, John E. Laird (AAAI)

2011

- Learning to Use Episodic Memory
  Nicholas A. Gorski, John E. Laird (Cognitive Systems Research)

2012

- Enhancing Intelligent Agents with Episodic Memory
  Andrew M. Nuxoll, John E. Laird (Cognitive Systems Research)
- A Multi-Domain Evaluation of Scaling in a General Episodic Memory
  Nate Derbinsky, Justin Li, John E. Laird (AAAI)