Episodic Memory (EpMem)

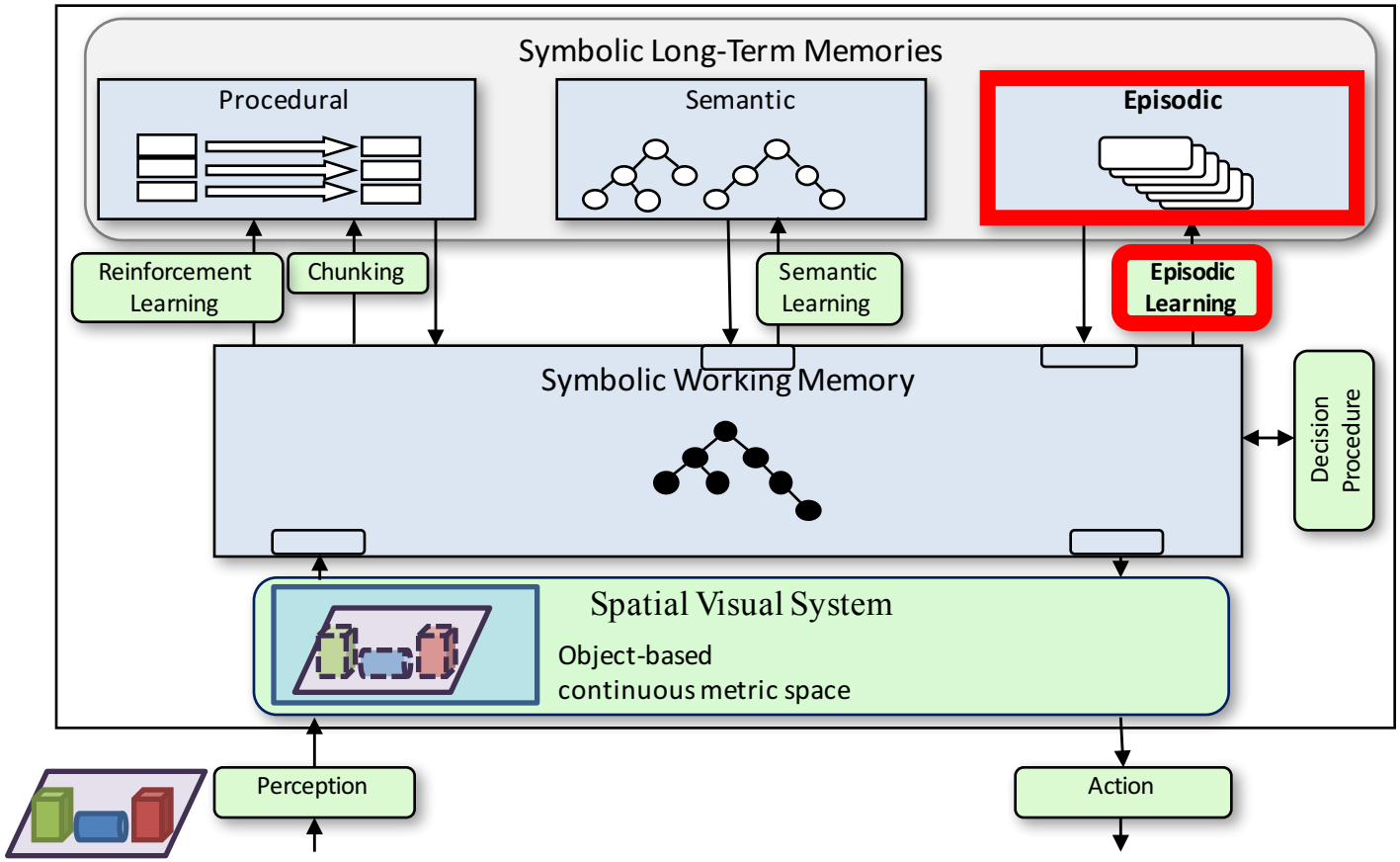
IJCAI 2016

Nate Derbinsky

Agenda

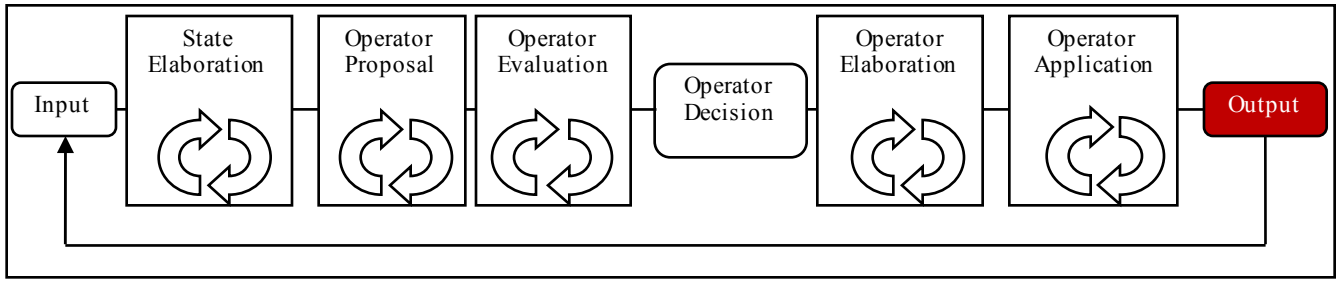
- Big picture
- Basic usage
- Example agents

Soar 9


Symbolic Long-Term Memories
Procedural
Semantic
Episodic
Reinforcement Learning
Chunking
Semantic Learning
Episodic Learning
Symbolic Working Memory
Decision Procedure
Spatial Visual System
Object-based
continuous metric space
Perception
Action

Soar Basic Functions

1. Input from environment
2. Elaborate current situation: *parallel rules*
3. Propose operators via acceptable preferences
4. Evaluate operators via *preferences: Numeric indifferent preference*
5. Select operator
6. Apply operator: Modify internal data structures: *parallel rules*
7. Output to motor system [and access to long-term memories]

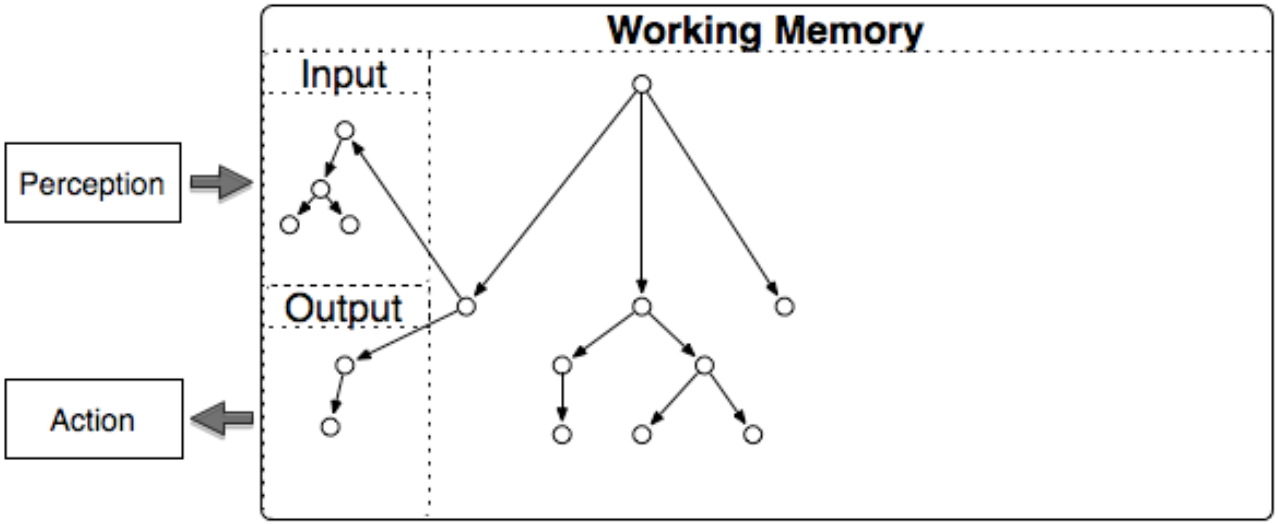


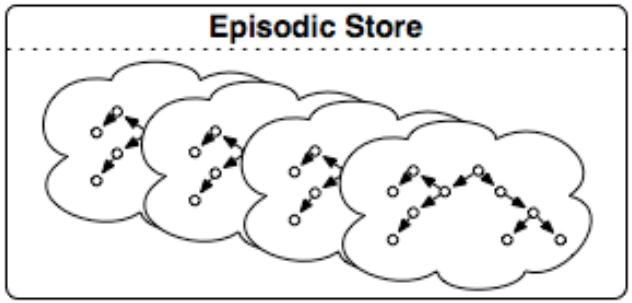
Episodic Memory: Big Picture

Episodic memory is a weak learning mechanism

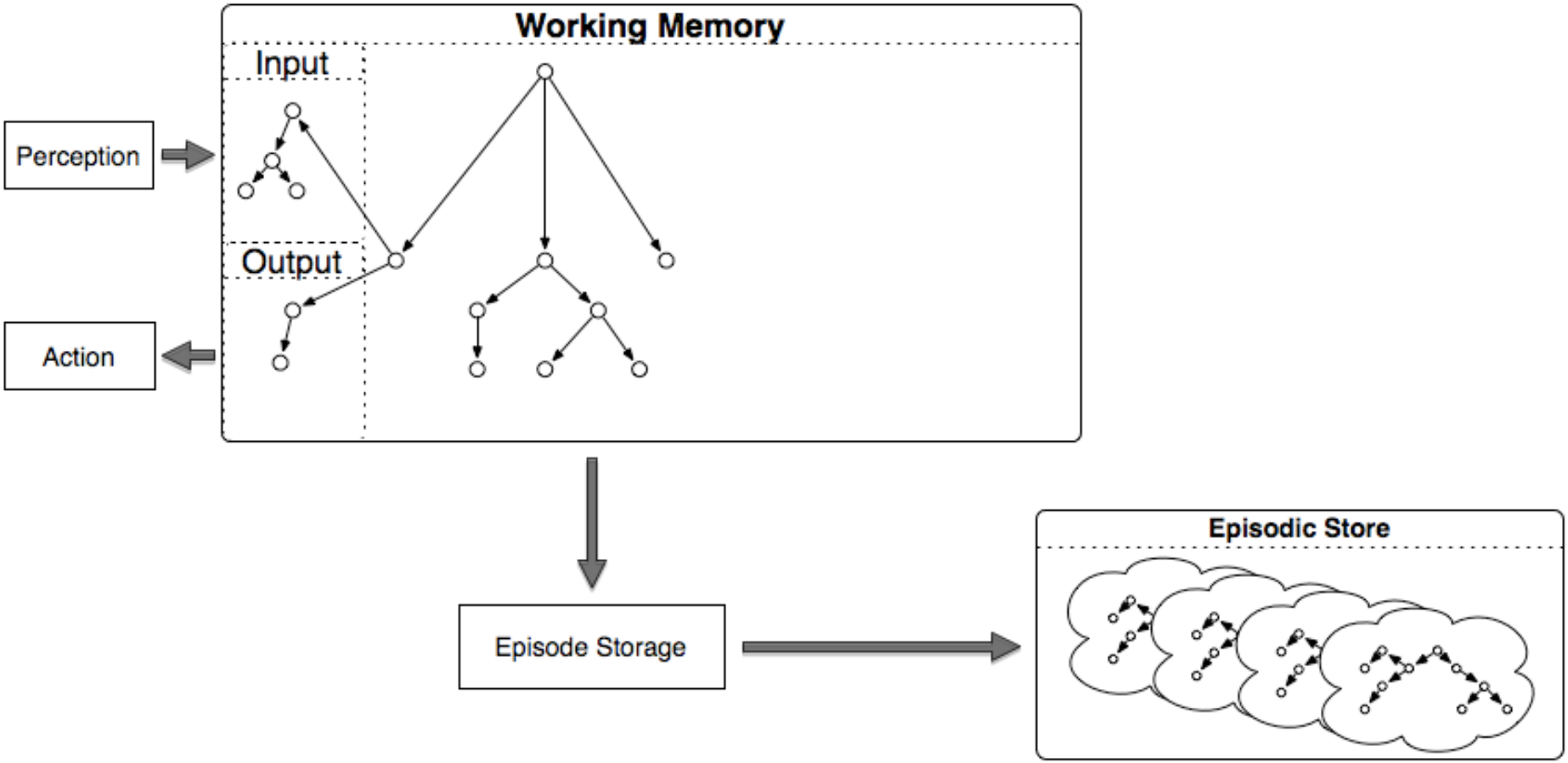
- Automatically captures, stores, and temporally indexes agent state
- Supports content-addressable agent interface to autobiographical prior experience

Architectural Integration

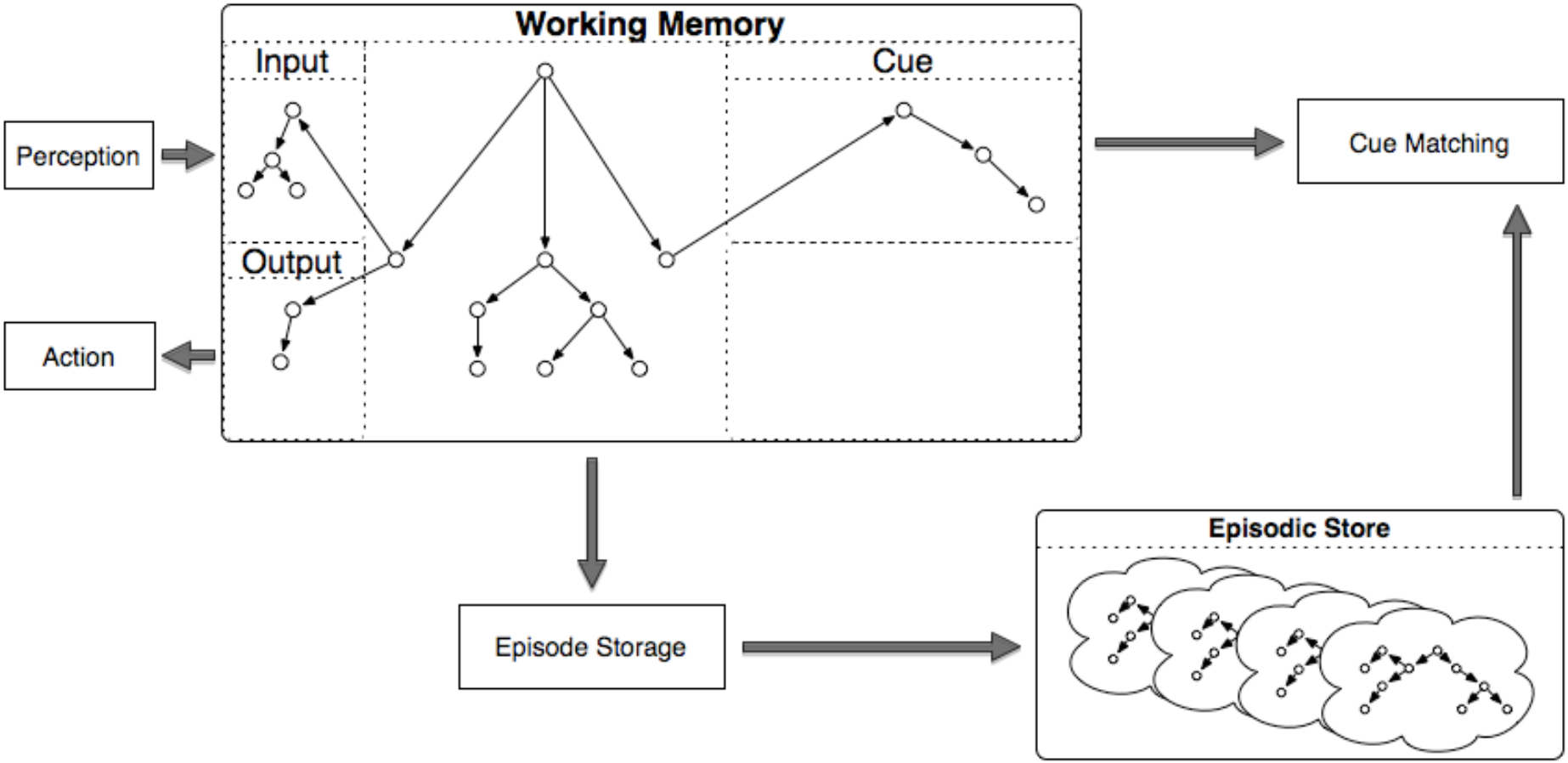




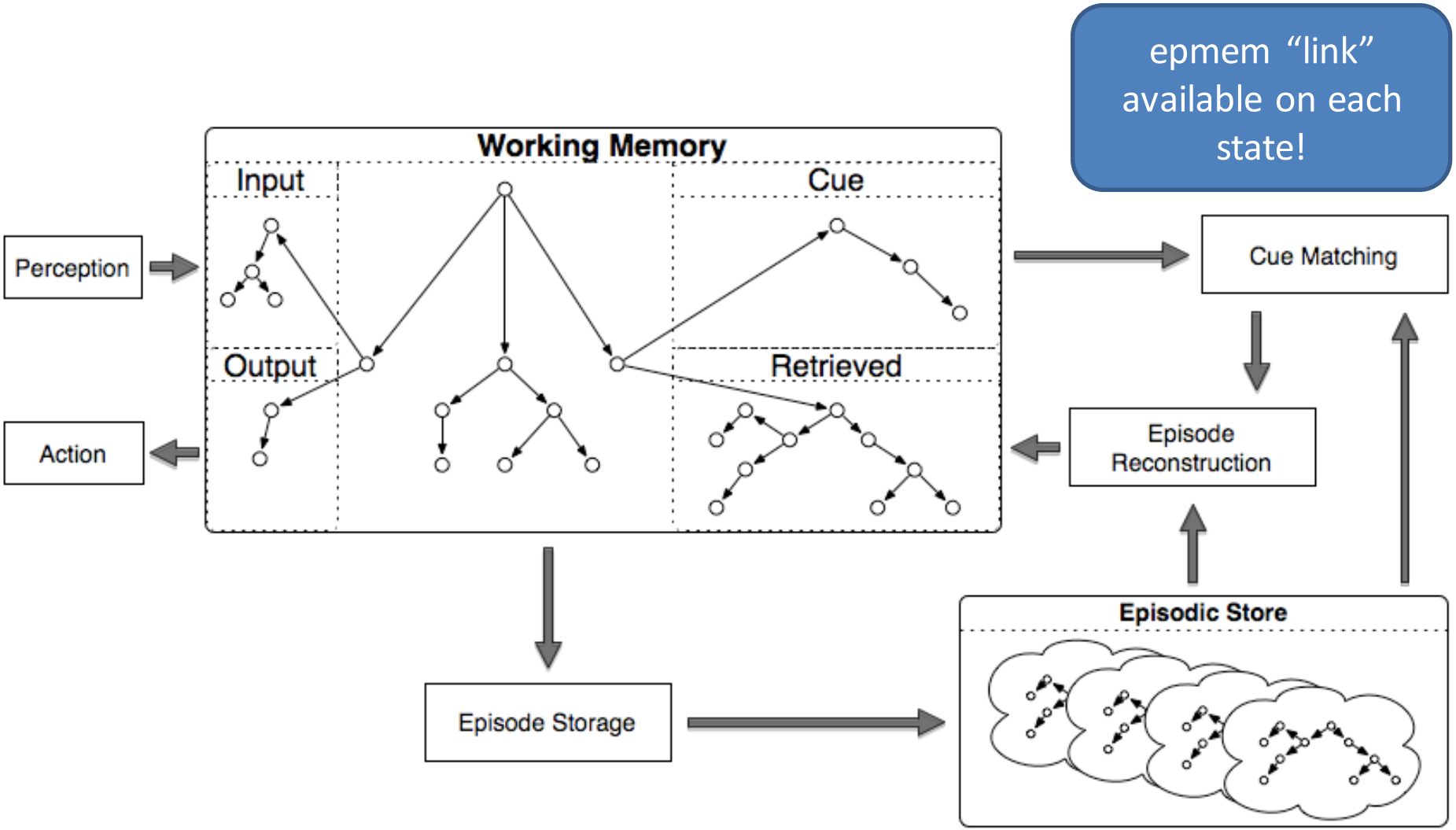
Architectural Integration



Architectural Integration



Architectural Integration



Basic Usage

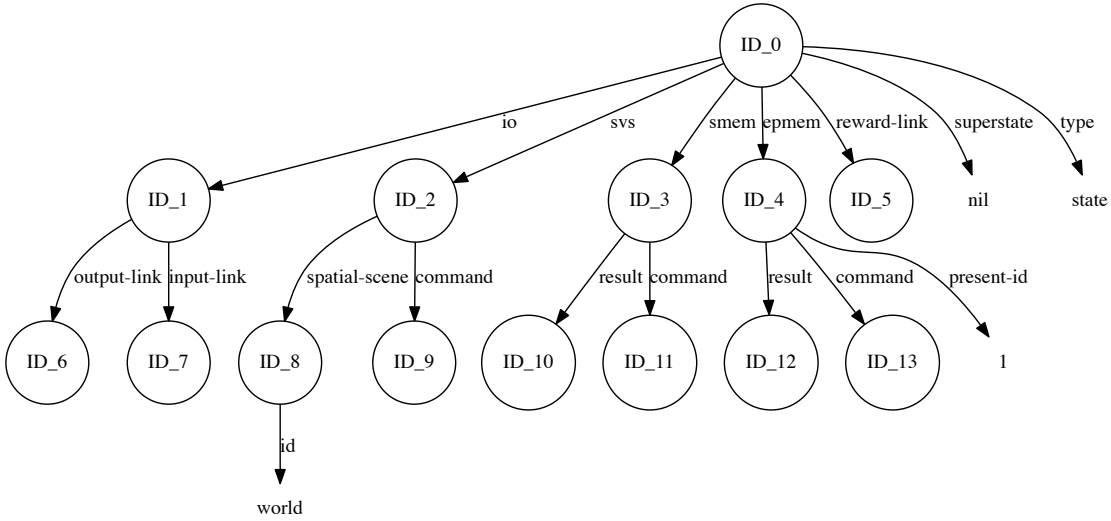
- Episodic-memory representation
- Storing knowledge
- Retrieving knowledge

Episodic-Memory Representation

Similar to working memory: symbolic triples

- Structures within an episode are connected; separate episodes are disconnected

```
(<id0> ^epmem <id4>
       ^io <id1>
       ^reward-link <id5>
       ^smem <id3>
       ^superstate nil
       ^svs <id2>
       ^type state)
(<id1> ^input-link <id7>
       ^output-link <id6>)
(<id2> ^command <id9>
       ^spatial-scene <id8>)
(<id3> ^command <id11> ^result <id10>)
(<id4> ^command <id13> ^present-id 1 ^result <id12>)
(<id8> ^id world)
```



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 (default)
    - `output`: new augmentation of output-link
- 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

Retrieving Knowledge

Cue-Based

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

Temporal Progression

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

Non-Cue-Based

Add an episode to working memory from episode #

Common Constraints:

- Requires that EpMem is enabled
- Only one per state per decision
- Processed during `phase`
- 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 acyclic
- 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

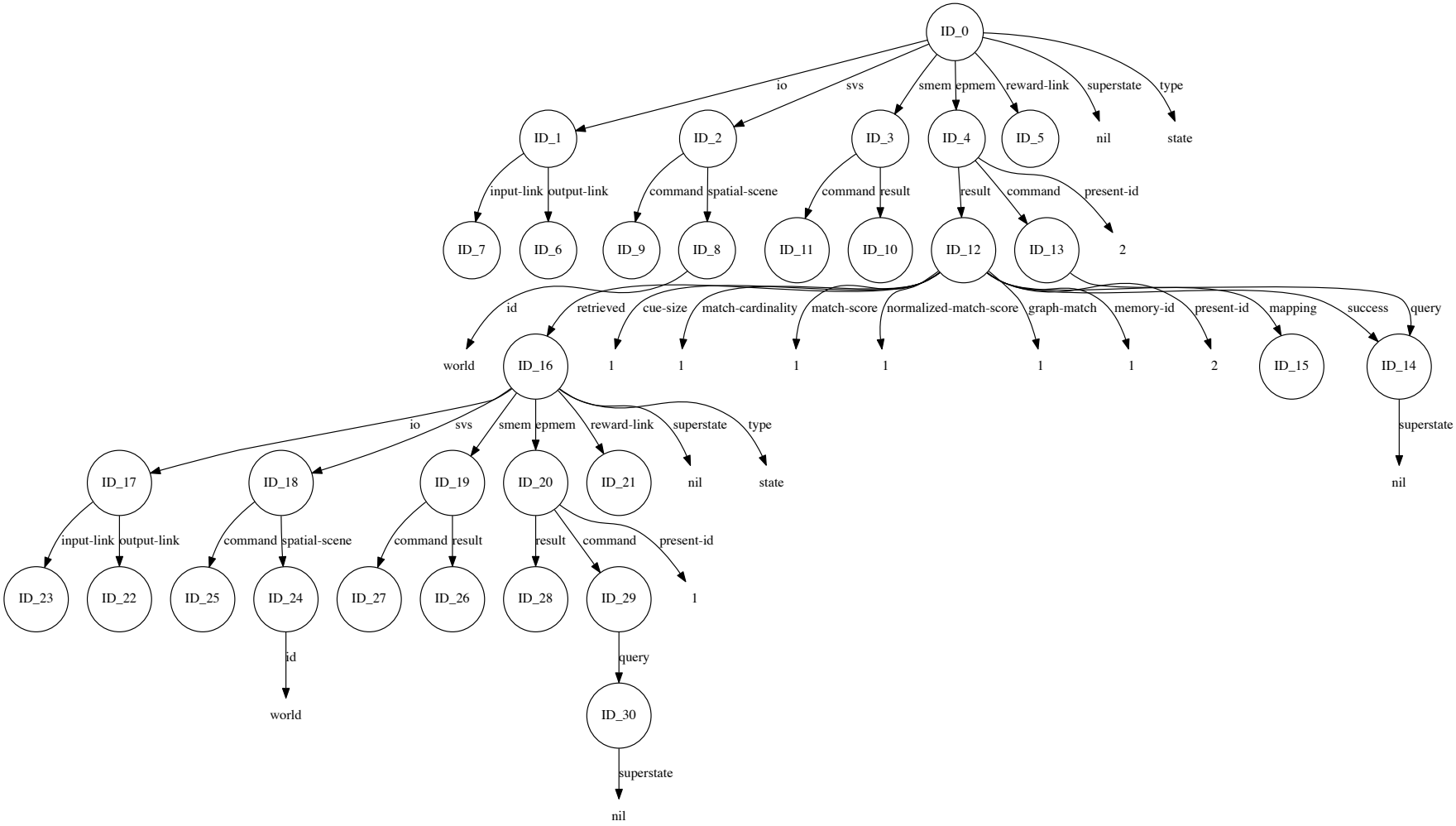
Values of cue WMEs are interpreted by type

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

Cue matching will return the most recent graph-matched episode, or the most recent non-graph-matched candidate episode with the maximal episode score (w.r.t. root-to-leaf path)

Cue-Based Retrieval: Example

Result



Cue-Based Retrieval

Optional Modifiers

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

Hard constraints on the episodes that can be retrieved.

Temporal Progression

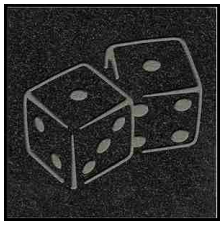
```
(<cmd> ^next <new-id>)
(<cmd> ^previous <new-id>)
```

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

EpMem Task: Virtual Sensing

epmem-virtual-sensing.soar

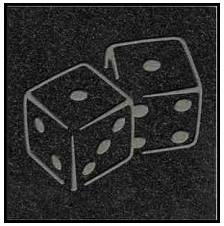
1. Produce a random number in WM
   EpMem automatically records this episode



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



Eaters!