

Efficiently Implementing Episodic Memory

Nate Derbinsky
University of Michigan

July 14, 2009

1

Computer Science and Engineering at Michigan



What is Episodic Memory?

- Long-term, contextualized store of specific events
 - Tulving, E.: Elements of Episodic Memory (1983)
- Functionally:
 - Architectural
 - Automatic
 - Autonoetic
 - Temporally indexed



July 14, 2009


2

Computer Science and Engineering at Michigan



CSE

The Promise of EpMem




MEMENTO

- Virtual Sensing
- Action Modeling
- Retroactive Learning
- ...

Nuxoll, A.: Enhancing Intelligent Agents with Episodic Memory. (2007)

July 14, 2009

3

Computer Science and Engineering at Michigan 

CSE

Efficient Implementation

Goals


- Develop a system that is practical for real-world tasks
- Establish baseline results for graph-based, task-independent EpMem implementations

Assumptions

- Stored episodes do not change over time
- Qualitative Nearest Neighbor (NN) cue matching

July 14, 2009

4

Computer Science and Engineering at Michigan 

Performance Challenges

- Consider a year of episodic memories...
 - 16 hours/day -> 42M to 420M episodes
 - 100 – 1000 features/episode (10-100 bytes/feature)
 - **21GB to 21TB**
 - 2GHz CPU -> **10 seconds/scan**

Laird, J.E., Derbinsky, N.: A Year of Episodic Memory (2009)

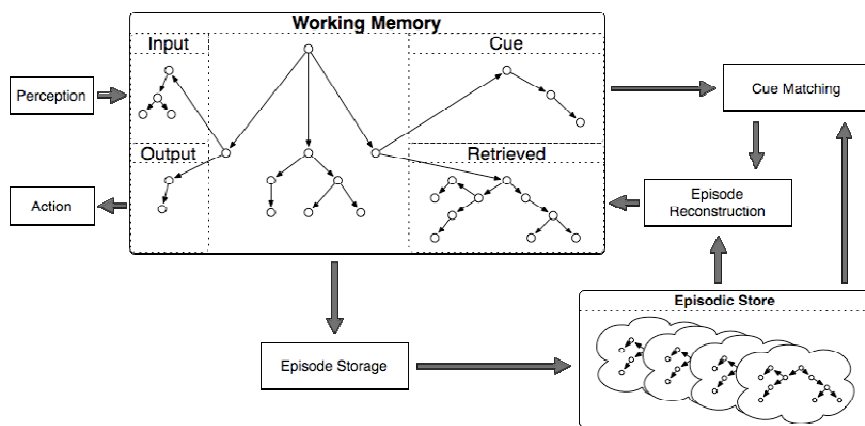
July 14, 2009

5

Computer Science and Engineering at Michigan



Integrating EpMem with Soar



July 14, 2009

6

Computer Science and Engineering at Michigan



CSE

Episodic Storage

- Faithfully capture Soar's top state of working memory
- Incrementally update indexing structures to facilitate efficient cue matching
- Minimize
 - Memory (monotonically increasing store)
 - Time (relatively frequent operation)




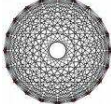



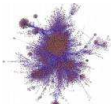




July 14, 2009

7

Computer Science and Engineering at Michigan

CSE

Episodic Storage: Naïve Implementation

Time	Working Memory		Episodic Store		
1			1 		
2			1 	2 	
3			1 	2 	3 

July 14, 2009

8

Computer Science and Engineering at Michigan

CSE

Compression via Global Memory Structure

- Observation
 - Agents tend to re-use WM structures
- Result
 - Maintain a global record of unique structures
 - Define episodes as “bag of pointers”

July 14, 2009

9

Computer Science and Engineering at Michigan

CSE

Gains via Interval Representation

- Observation
 - An episode will differ from the previous (and next) only in a relatively small number of features
- Result
 - Define episodes implicitly as temporal changes

July 14, 2009

10

Computer Science and Engineering at Michigan

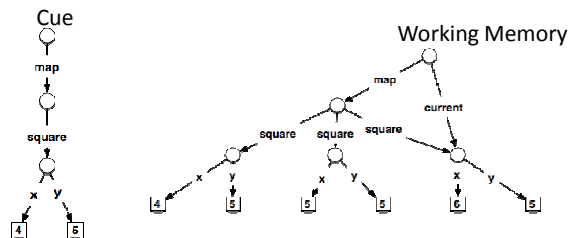
Episodic Storage Summary

- Maintain record of unique WM structures
- Maintain associated intervals on WME addition/removal
 - *Only process changes!*

Episodic storage performs in time/space linear in the *changes* in working memory.

Cue Matching

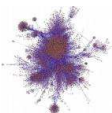
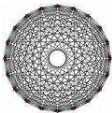

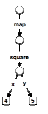




- A cue is an acyclic graph, partially specifying a subset of an episode



- Cue matching returns the most recent episode containing the greatest number of cue leaf elements


CSE

Cue Matching: Naïve Implementation

Time	N	N-1	N-2
Episode			
			
Score	ξ	Ω 	β

July 14, 2009

13

Computer Science and Engineering at Michigan 

CSE


Minimizing Combinatorics via Two-Stage Matching

1. Evaluate *candidate* episodes based upon relatively inexpensive surface match
2. Perform combinatorial structural match (graph-match via CSP backtracking) **ONLY** on candidate episodes with a perfect surface score

End search on perfect match or no more episodes.

July 14, 2009

14

Computer Science and Engineering at Michigan 

CSE Minimize Episode Evaluation via Interval Endpoint Search

Episode match score changes only at interval endpoints!

July 14, 2009 15 Computer Science and Engineering at Michigan

CSE Interval Search Model

Interval Search Operation Time (s)

$y = 1E-06x + 0.0117$
 $R^2 = 0.9888$

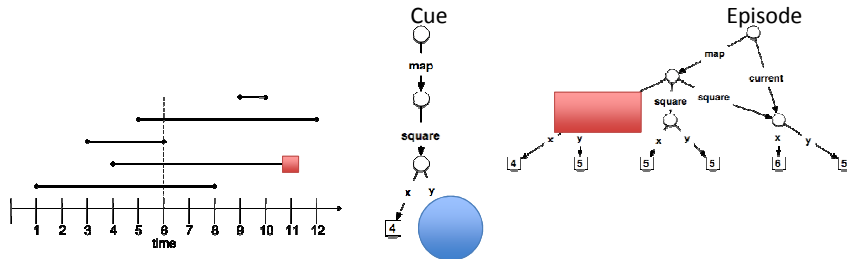
$(1/T)(Distance)(\Delta)$

- T** = Total episodes
- Distance** = Temporal distance to best match
- Δ** = Cue intervals (~ WM changes)

Interval search is dependent upon the number of candidate episodes evaluated.

July 14, 2009 16 Computer Science and Engineering at Michigan

Efficient Surface Evaluation via Incremental DNF Satisfaction



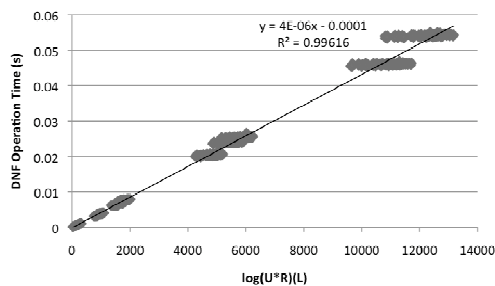
- $\text{sat}(y=5) := (\text{root AND map}[1] \text{ AND } \text{AND } y=5[1]) \text{ OR } (\text{root AND map}[1] \text{ AND square}[2] \text{ AND } y=5[2]) \text{ OR } (\text{root AND map}[1] \text{ AND square}[3] \text{ AND } y=5[3])$
- Surface matching can be expressed as evaluating the satisfaction of a set of disjunctive normal form (DNF) Boolean equations
 - Each interval endpoint inverts the value of a single variable

July 14, 2009

17



DNF Model



- U** = Unique nodes
- R** = Stored intervals (~ changes)
- L** = Cue node literals

DNF performance is dependent upon the changes in working memory.

July 14, 2009

18



CSE

Cue Matching Summary

- Minimize candidates by only considering episodes with at least one cue node
- Minimize combinatorics via two-stage matching policy
 - Exponential growth in the worst case
- Minimize episode evaluation via interval endpoint search
 - Linear growth in the worst case
- Minimize surface evaluation cost by only processing cue node changes

July 14, 2009
19
Computer Science and Engineering at Michigan

CSE

Episode Reconstruction

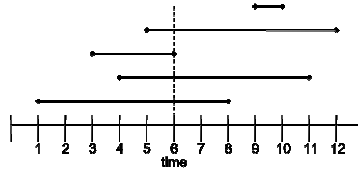
- The process of faithfully reproducing all episode content and structure within the agent's working memory
 - Collect contributing episode elements
 - Add elements to working memory

July 14, 2009
20
Computer Science and Engineering at Michigan

CSE

Logarithmic Interval Query via Relational Interval Tree

- Collecting episode elements in an Interval representation is tantamount to an interval intersection query:
 - Collect all elements that started before and ended after time t



- By implementing an interval tree, intersection queries are answered in time logarithmic with respect to the changes in working memory

July 14, 2009

21

Computer Science and Engineering at Michigan



CSE

Empirical Domain



- TankSoar Mapping-Bot*
 - 2500 features
 - 70-90% of perceptual WMEs change each episode
- 2.8GHz, 4GB RAM
- SQLite3

July 14, 2009

22

Computer Science and Engineering at Michigan



CSE

Empirical Results

1 million episodes (~1 episode/decision), 10 trials

Storage	Cue Matching*	Reconstruction**	Total
2.68ms 625-1620MB (0.64-1.66KB/ep)	57.6ms	22.65ms	82.93ms

* 15 cues
** 50 random times

July 14, 2009

23

Computer Science and Engineering at Michigan

CSE

Future Work

- **Better Evaluation**
 - Characterize architecture performance with respect to properties of the environment, agent, cues, and task
 - Longer and multi-task runs
- **Bound Cue Matching**
 - Fast familiarity
 - Heuristic graph-match
- **Algorithmic Variants**
 - Selection bias: activation, arousal via appraisals, etc.
 - Characterize task vs. architecture performance

July 14, 2009

24

Computer Science and Engineering at Michigan

CSE
Evaluation

Nuggets

- Extended and improved Soar-EpMem implementation (9.1.1)
- 1M episode initial empirical study with predictive performance models

Coal

- Limited evaluation

July 14, 2009
25
Computer Science and Engineering at Michigan

CSE
Further Reading

- Derbinsky, N., Laird, J.E.: *Efficiently Implementing Episodic Memory*. To Appear: Proceedings of the 8th International Conference on Case-Based Reasoning (2009)
- Laird, J.E., Derbinsky, N.: *A Year of Episodic Memory*. To Appear: Workshop on Grand Challenges for Reasoning from Experiences, 21st International Joint Conference on Artificial Intelligence (2009)

July 14, 2009
26
Computer Science and Engineering at Michigan