Effective and Efficient Management of Soar's Working Memory via Base-Level Activation

Nate Derbinsky and John E. Laird

University of Michigan

Our Focus

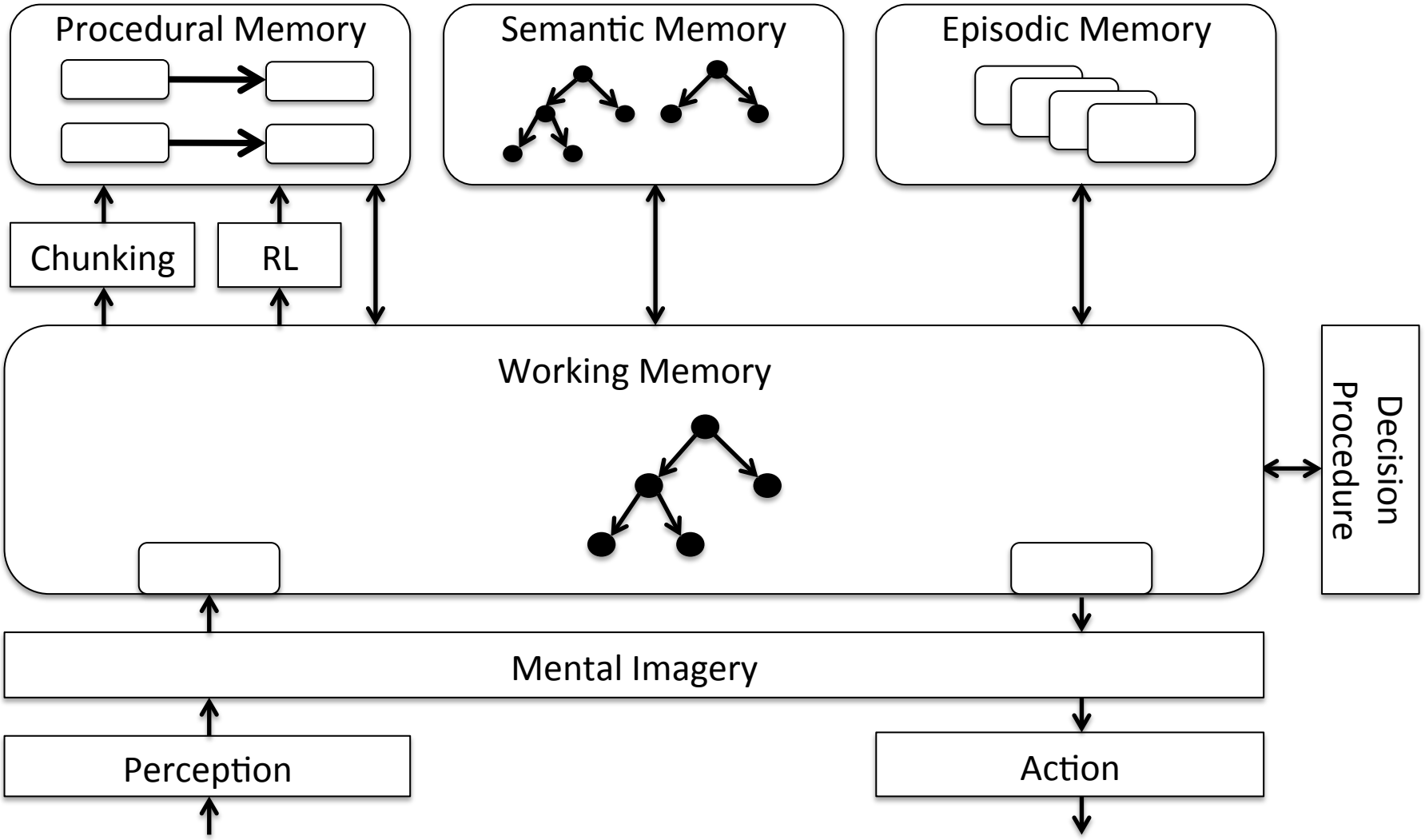
Human-Level Intelligence

- Long-living, learning, real-time (<50 msec.) agents
- Numerous, complex tasks/environments
- NOT modeling human behavior/processes

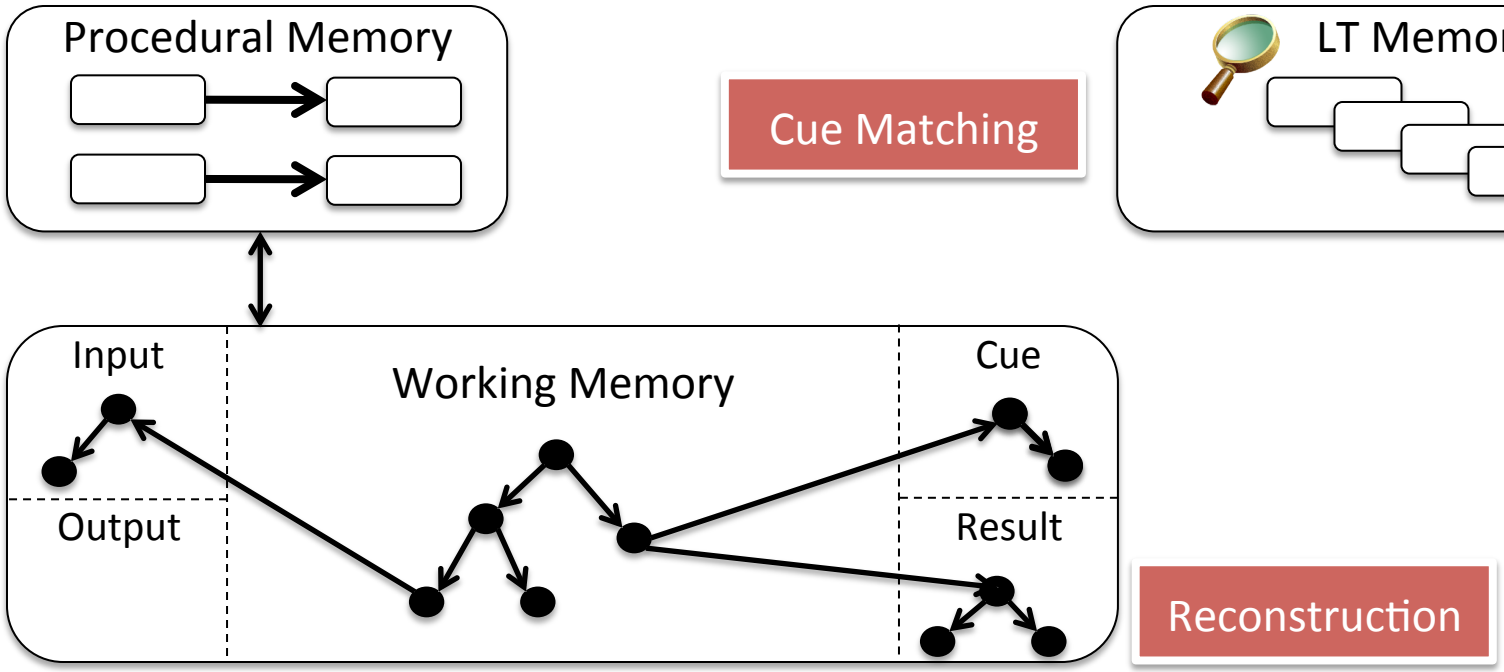
Cognitive Architecture

- Computational infrastructure for intelligent agents
- Aspects of cognition that are constant over time and across different application domains

The Soar Cognitive Architecture



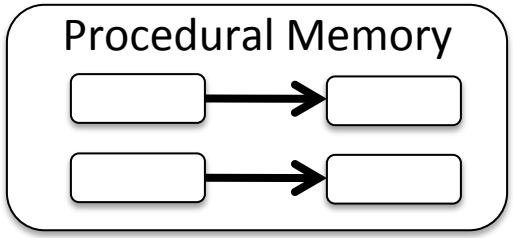
Soar: LT Memory Access

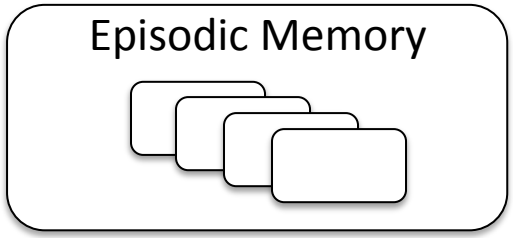


The reactivity of a Soar agent is the time required to make a decision, which includes accessing and modifying long-term memories

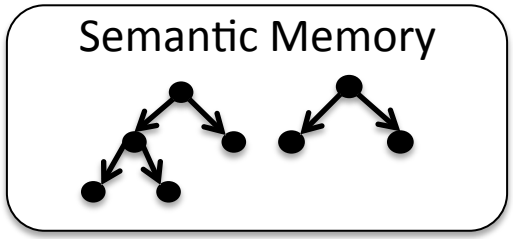
Small Working Memory

Necessity. Bounding Memory Retrievals

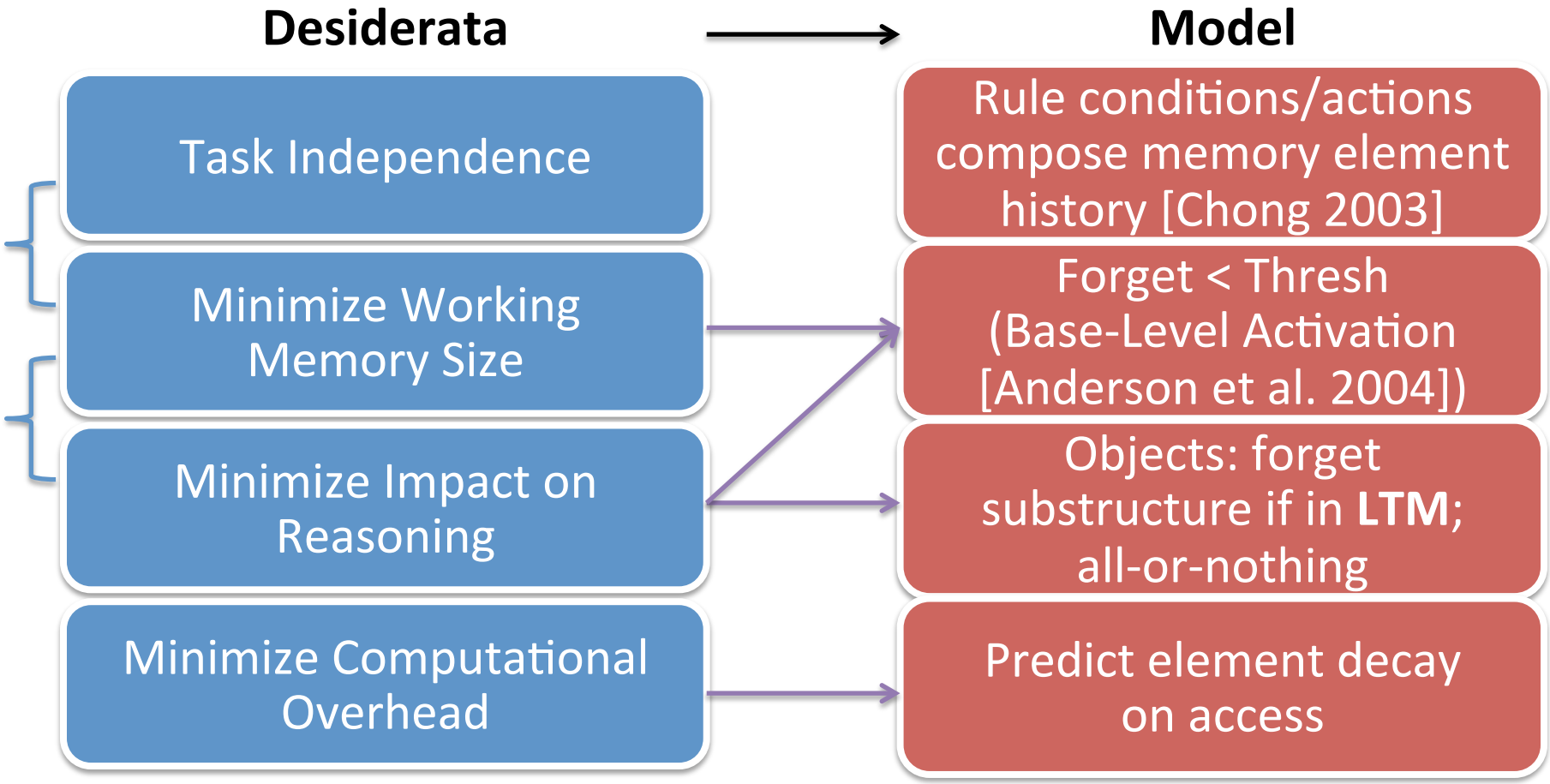




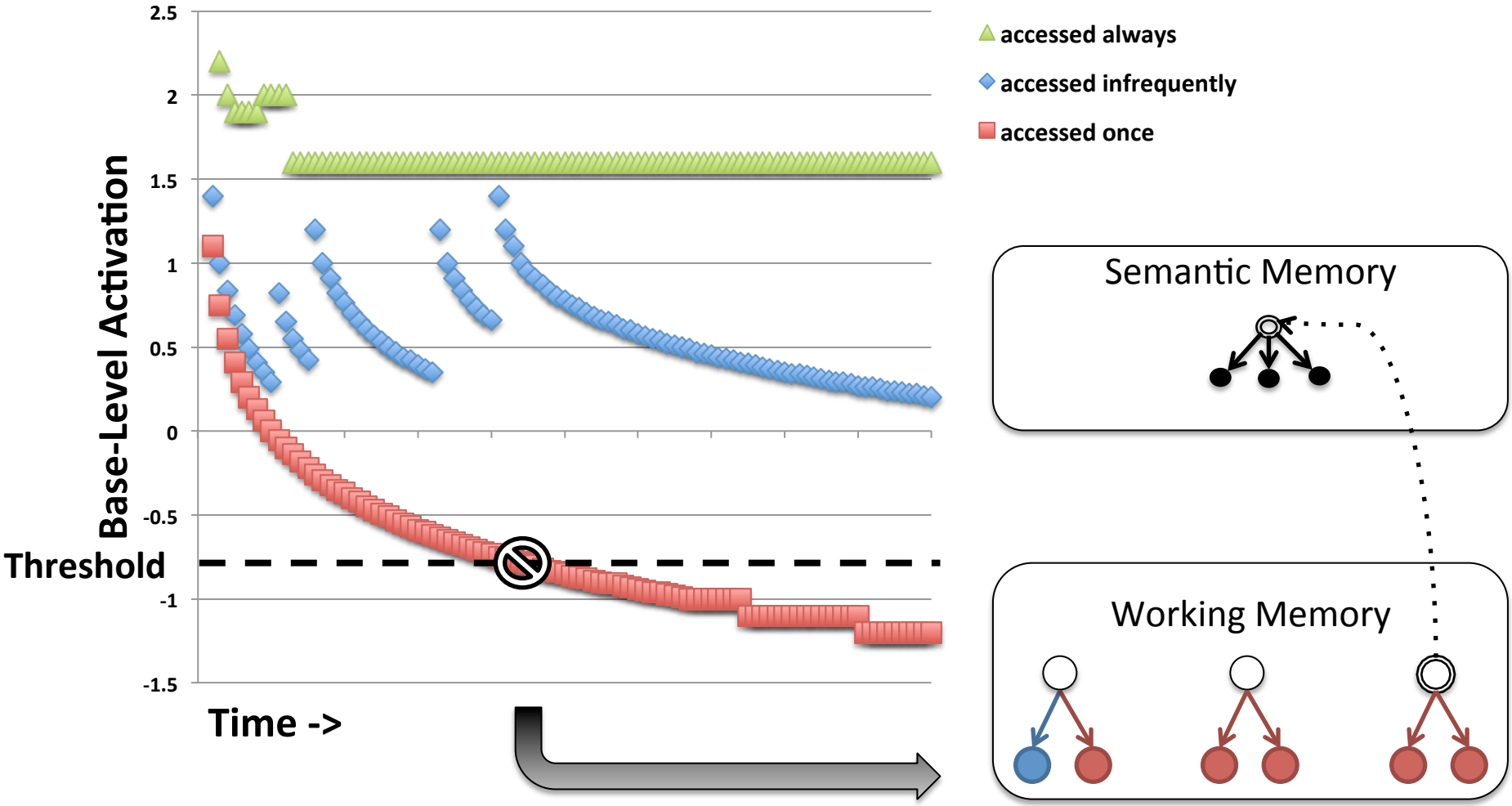
Opportunity. Efficient LT Memory Access



Working Memory Maintenance



Maintenance Model



Evaluating the Efficiency of a Maintenance Mechanism

- Number of elements in working memory (**N**)
  - Large memory: ↑**N**
- Number of activations/cycle (**E**)
  - Dynamic, knowledge-rich agent: ↑**E**
- Element lifetime (**L**)
  - Frequently accessed elements: ↑**L**
  - High element turnover: ↓**L**

Naïve Approach

Algorithm

- At each time step
  - For each memory element
    - If (Activation < Threshold)
      - » Forget

Efficiency Evaluation

- Per Time Step: O(**N**)
- Per Memory Element: O(**L**)

Our Approach: Decay Prediction

Algorithm ~ (Nuxoll et al. 2004)

- On new activation event
  - *Predict* time of future decay
  - Add to *time-step-keyed map*
- Each time step
  - Remove keyed decayed elements in map

Efficiency Evaluation

- Per Time Step: O(# decayed + **E***[Prediction Cost])

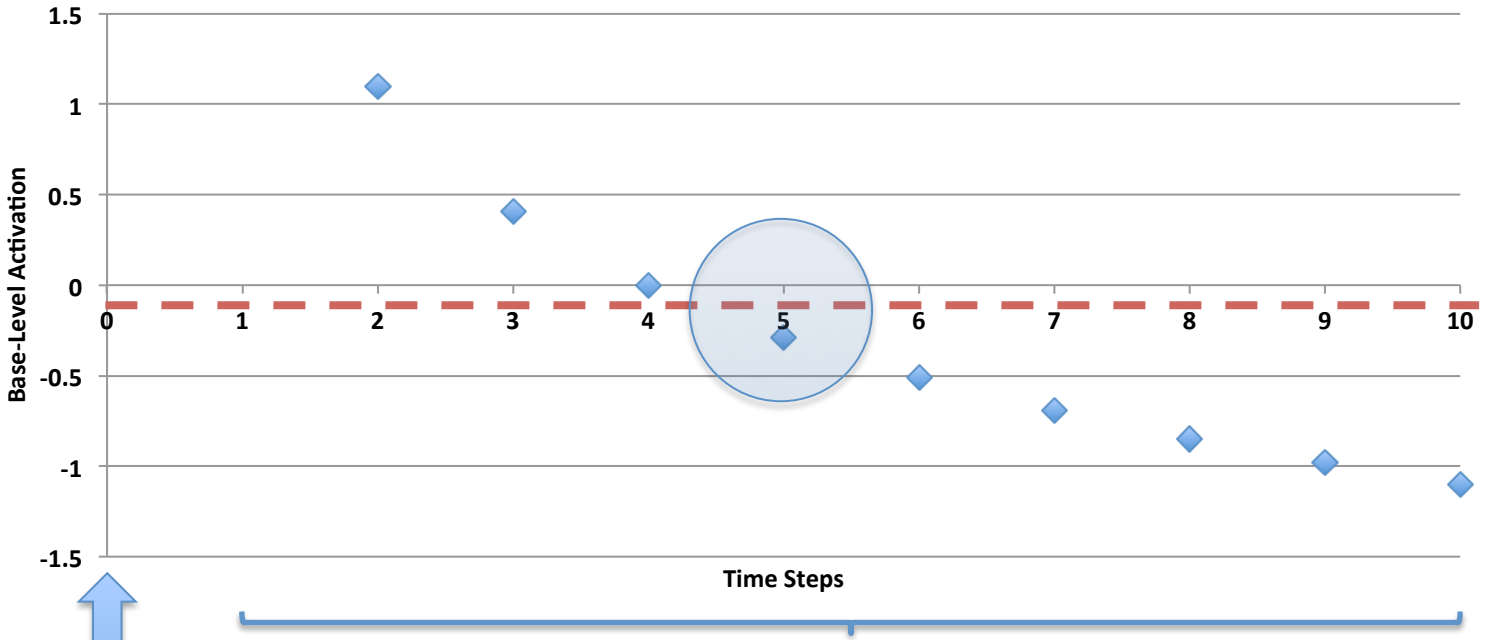
Decay Prediction

1. Cheaply approximate decay on each access
   - Underestimate time of decay by treating each memory access independently: O(1)

2. Exact determination
   - Binary parameter search: $O(\log_2 \mathbf{L})$
   - Not needed if element is removed by #1 estimate
   - Otherwise, reduced by the degree to which #1 is accurate

Decay Prediction

Example



Threshold
(θ)

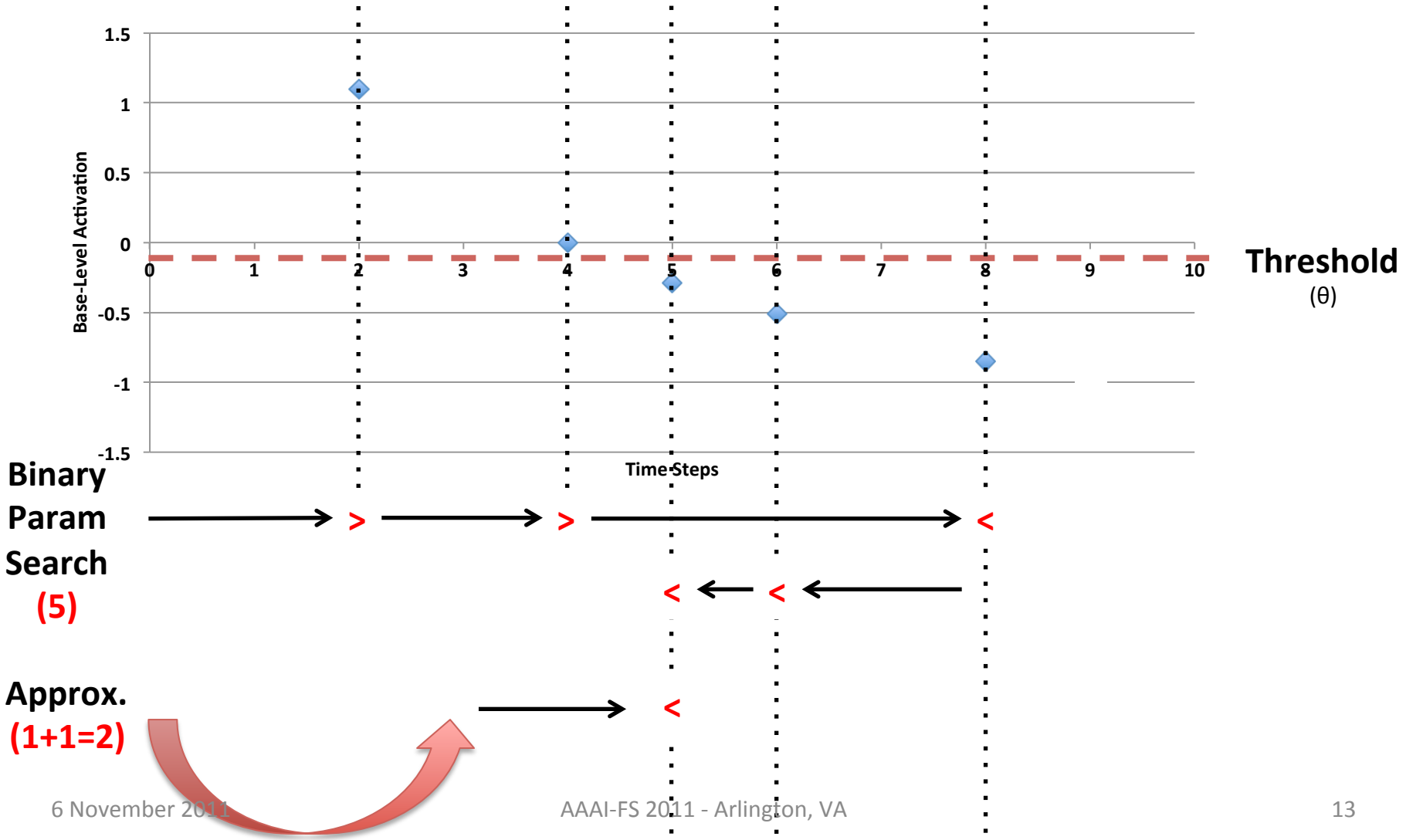
How far in the future will the memory decay?

Now

Given: decay rate (d), access history (t_1, t_2, t_3, …)

Decay Prediction

Example



Novel Base-level Decay Approximation

Given

constants

- Decay threshold (θ)
- Decay parameter value (*d*)

and a set of memory accesses…

- Time since access (*s*)
- Number of accesses (*n*)

solve for…

- Time till memory decay (*t*)

Algorithm

For each memory access…

$$\ln(n \cdot [t+s]^{-d}) = \theta$$

$$\ln(n) - d \cdot \ln(t+s) = \theta$$

$$\ln(t+s) = \frac{\theta - \ln(n)}{-d}$$

$$t = e^{\frac{\theta - \ln(n)}{-d}} - s$$

Decay estimate = $\sum t$

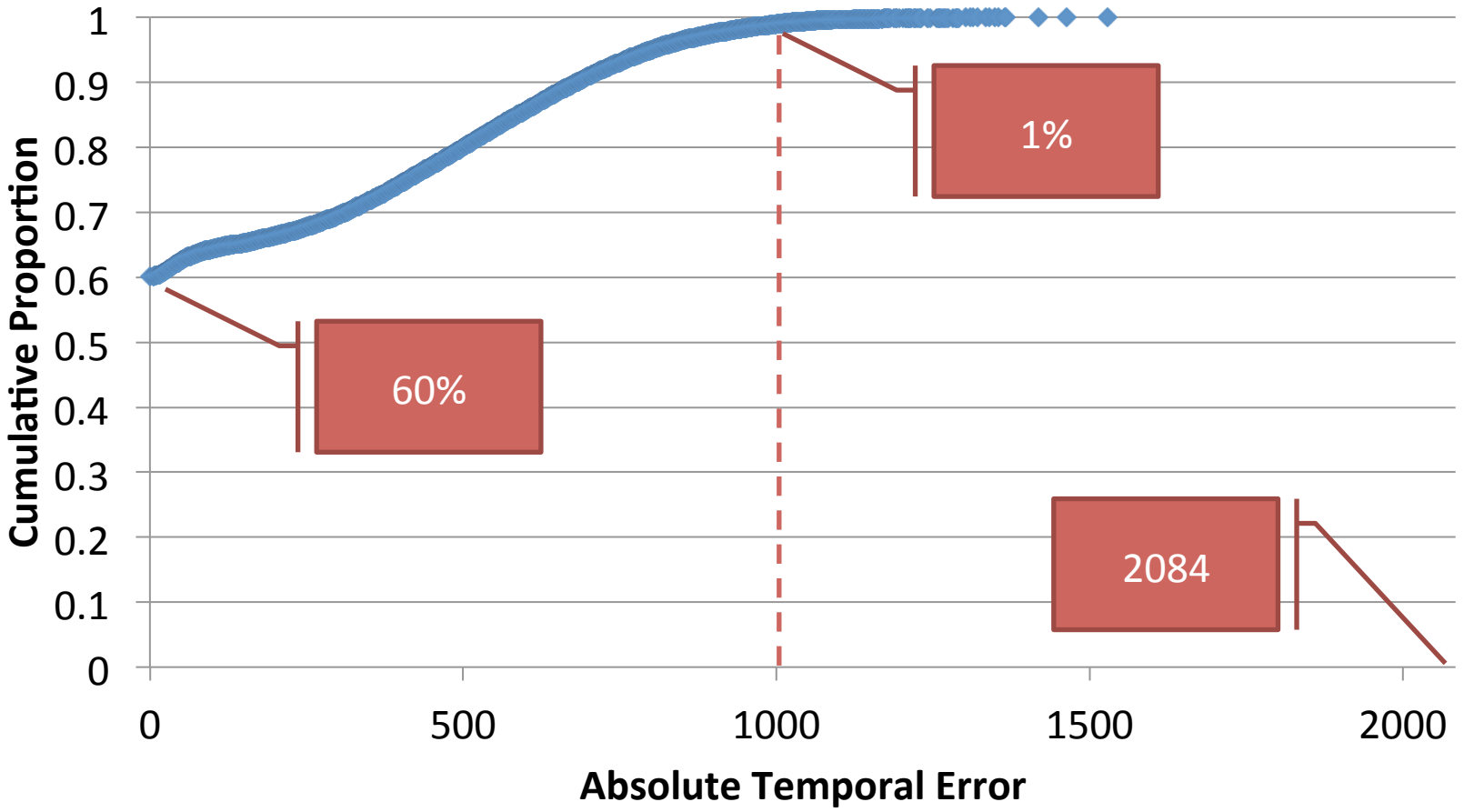
Mechanism Evaluation

1. **Synthetic**

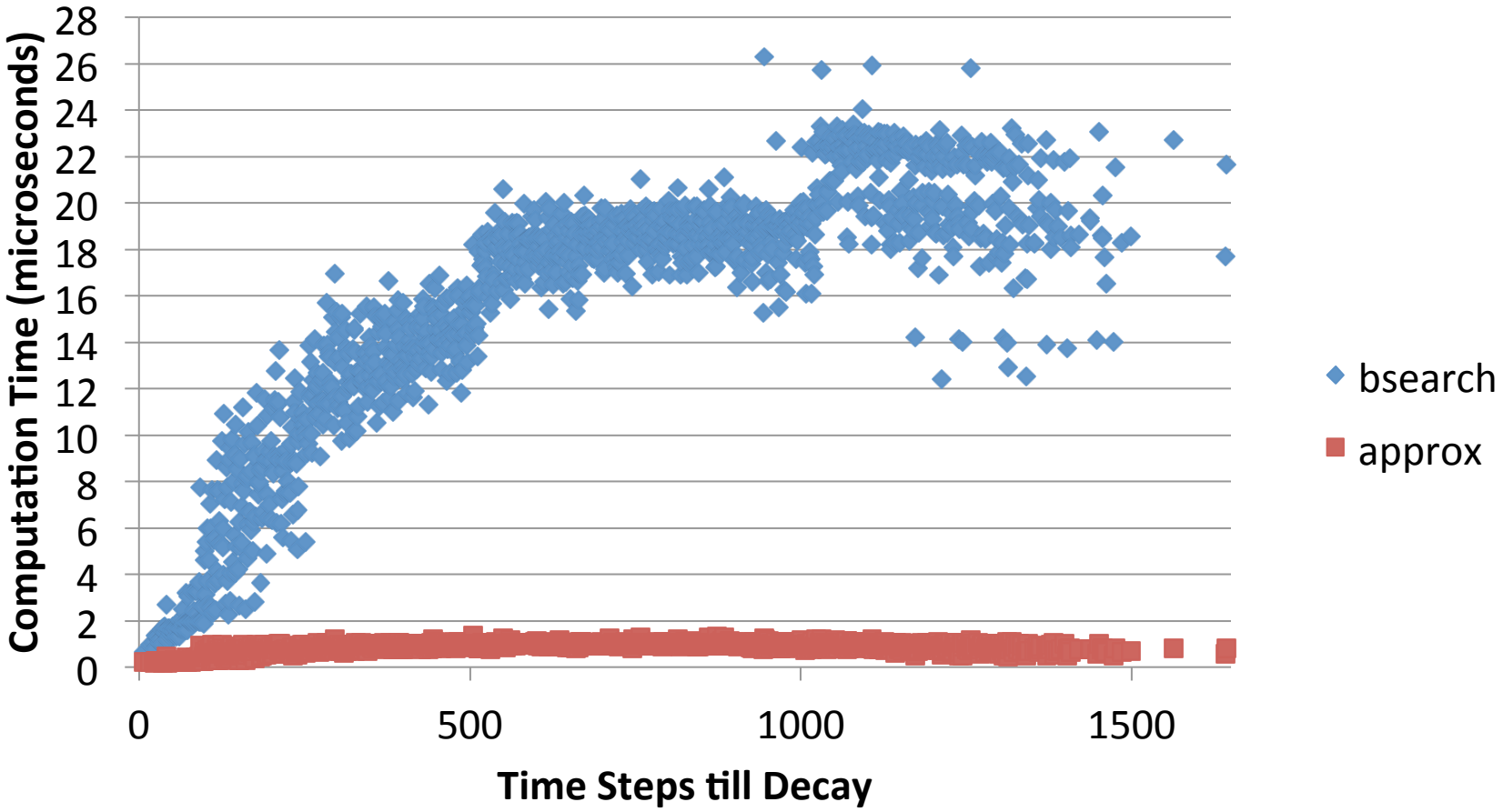
50k random, valid histories

2. **Mobile Robotics**

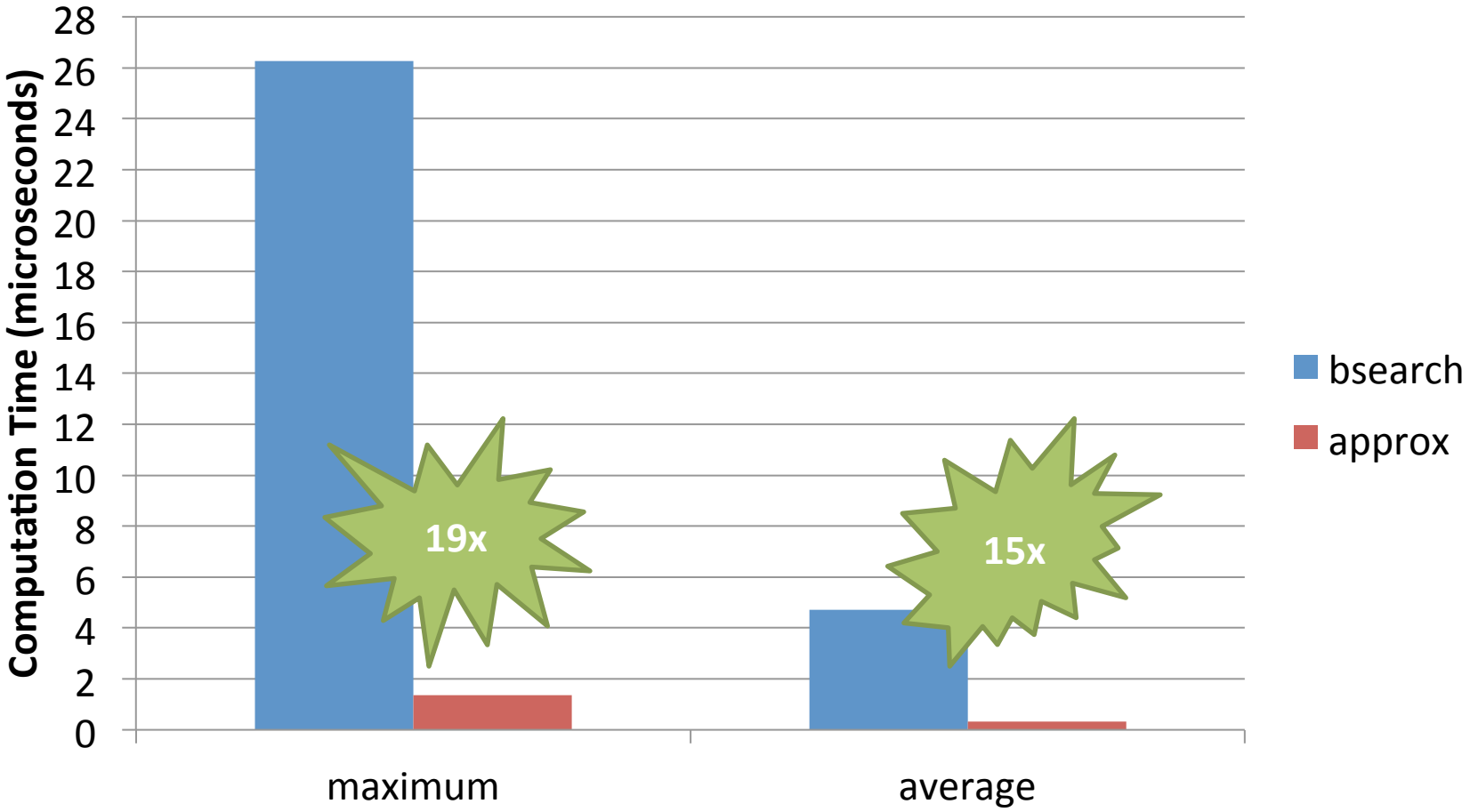
Synthetic: Approximation Quality



Synthetic: Prediction Complexity



Synthetic: Prediction Computation



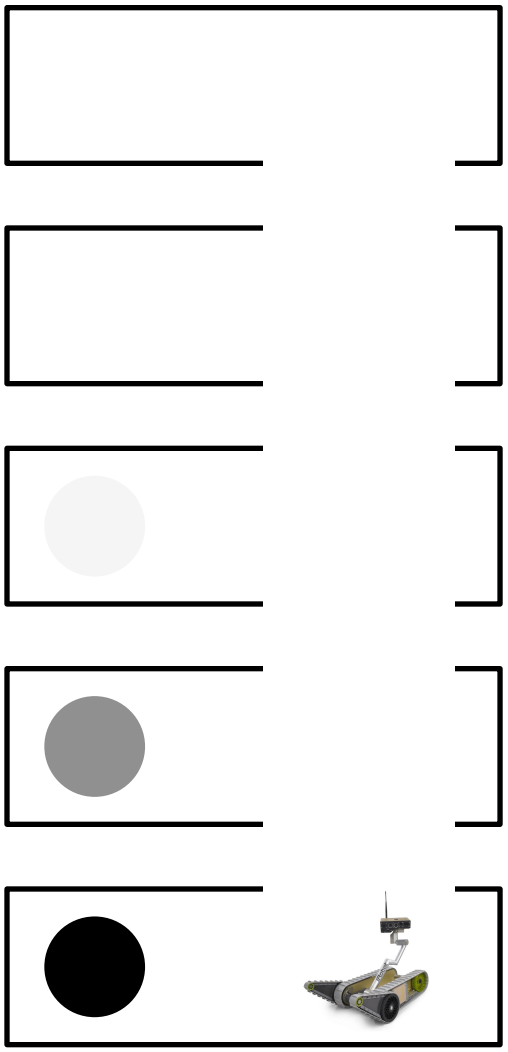
Mobile Robotics Task

In Simulation

Exploration

- 3rd floor, CSE Building, UM
  - 110 rooms
  - 100 doorways
- Builds map in memory from experience

Agent Map Knowledge



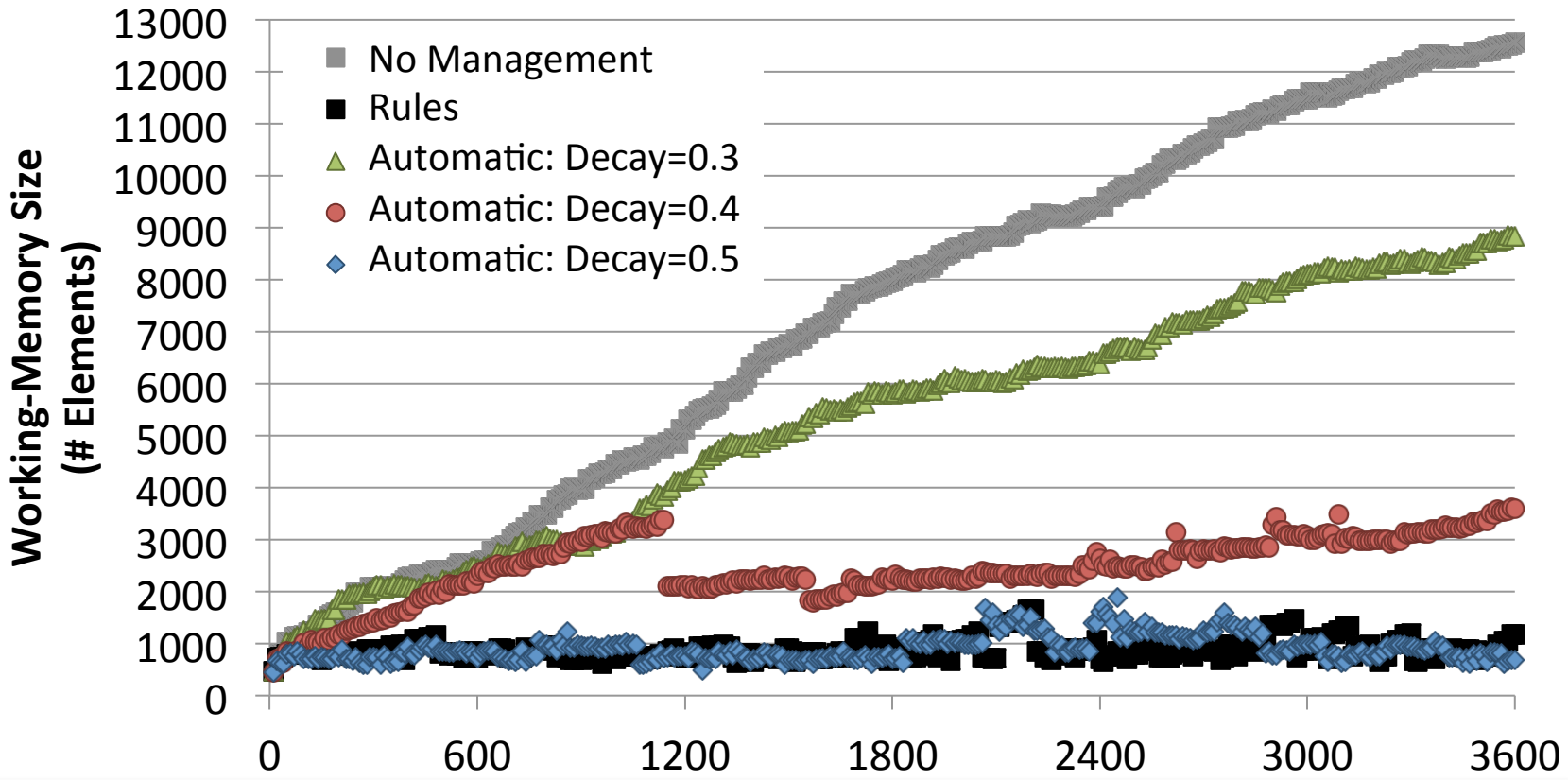
Room Features

- Position, size
- Walls, doorways
- Objects
- Waypoints

Activation

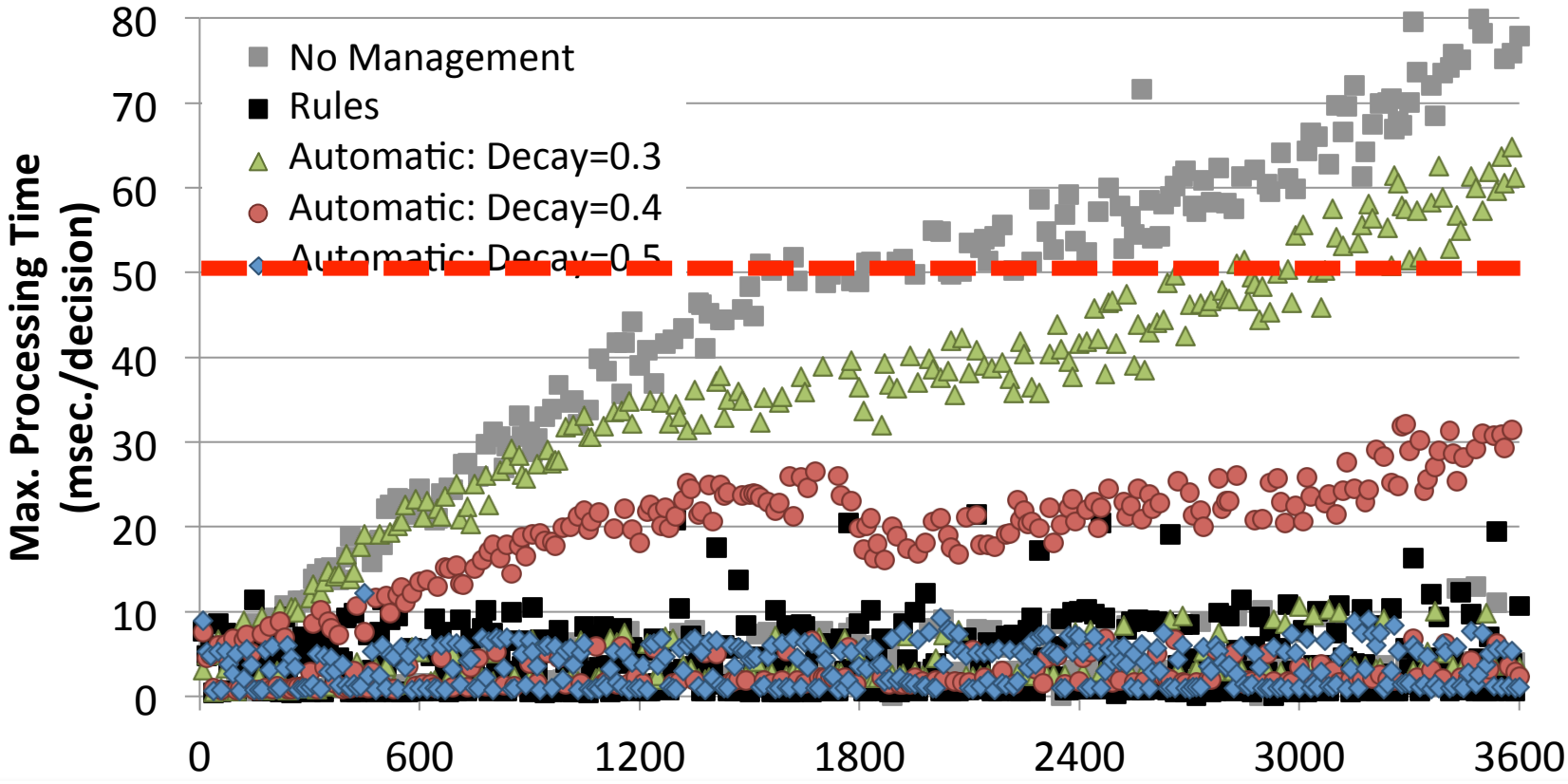
- Initial exploration
- Planning/Navigation

Mobile Robotics Data (1)



With appropriate decay, base-level activation maintains working memory size comparably with task-specific rules

Mobile Robotics Data (2)



With appropriate decay, base-level activation maintains reactivity comparably with task-specific rules

Automatic WM Maintenance

- Takes advantage of multiple memory systems
- Improves agent reactivity
  - Low computational overhead
  - Sound agent reasoning
  - No task-specific knowledge

Related Work

Modeling

- ACT-R [Anderson et al. 2004]
- Soar [Chong 2003][Chong 2004]

Cognitive Benefits

- Task Switching [Altmann and Gray 2002]
- Heuristic Inference [Schooler and Hertwig 2005]
- **Agent Reactivity [This Work]**

Limitations

- Single decay model
  - Are recency/frequency sufficient to capture memory element importance?
- Single task
  - Regularities of spatial locality map well to recency/frequency
  - LTM elements are never inconsistent with WM

Thank You :-)

Questions?