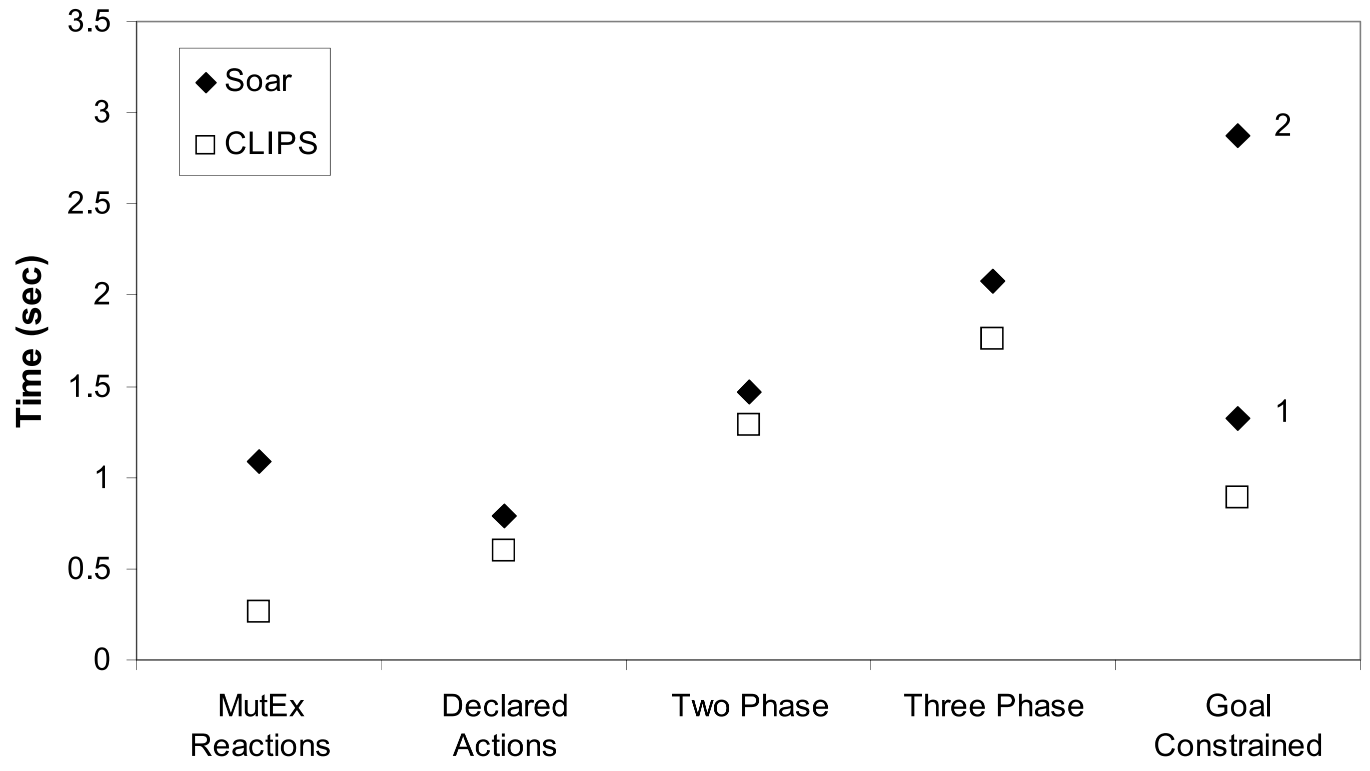
AI Architecture Evaluation and the Soar-Lite Project

Scott Wallace & John Laird
University of Michigan
Artificial Intelligence Lab

Last Year

- We performed an initial comparison of the Soar and CLIPS architectures
  - Examined problems in Towers of Hanoi and Eaters
  - Found both quantative and qualitative differences in the performance of these architecture
  - Discovered that in some situations, Soar’s native subgoaling mechanism was expensive

Previous Results



Modularized Features

- Compile time flags are used to determine what modules to include
  - Detailed Timing Facilities
  - High-cost Callbacks
  - Learning/Justifications
  - Backtracing/GDS Support
  - And More…

Detailed Timing Facilities

- Removes all the detailed timers
- Retains
  - total cpu time
  - total kernel time
  - phase timers
- Related Modules
  - Kernel Time Only
    - retain only total kernel time and total cpu time

High Cost Callbacks

- Removes the majority of callbacks issued during Soar’s execution
  - ~20 Runtime Callbacks
- Retains:
  - Initialization callbacks
  - Input Cycle callback
  - Output Cycle callback
  - After Decision Phase callback

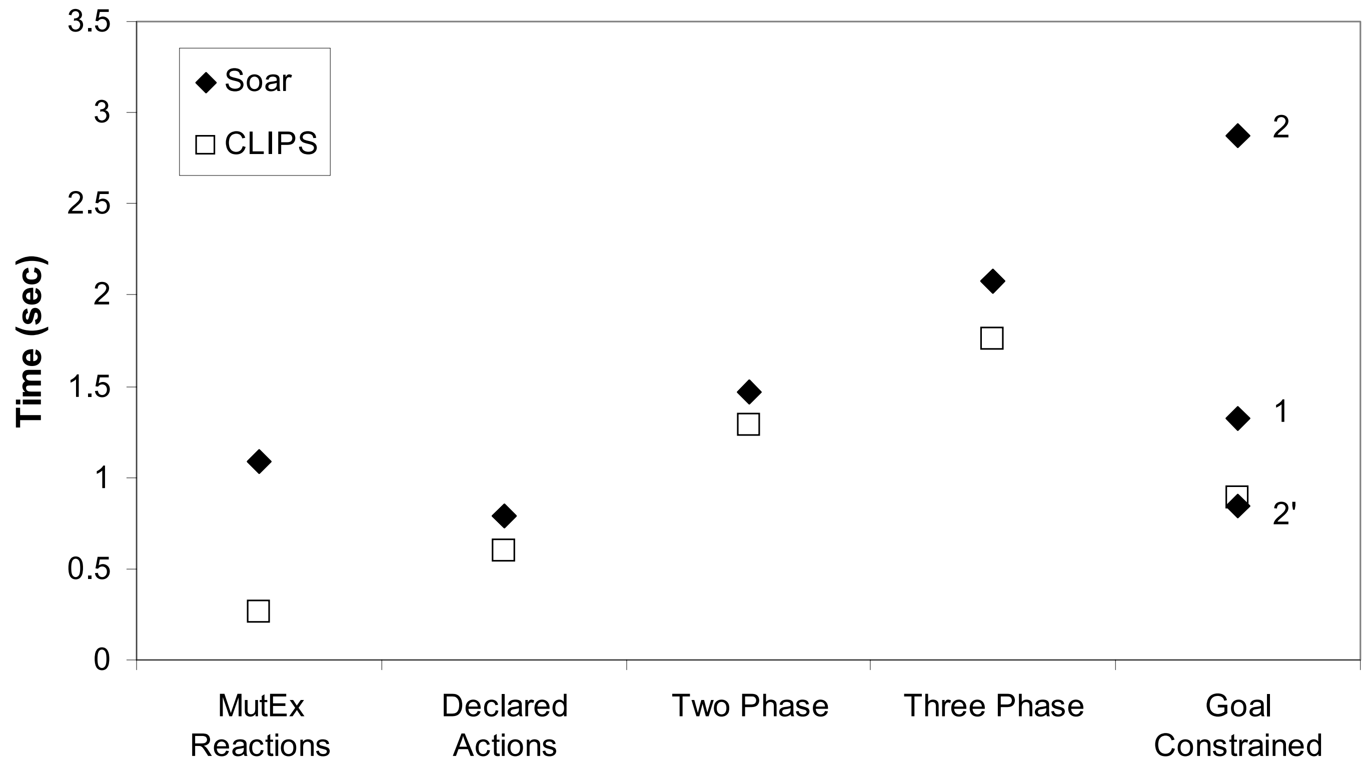
Learning/Justifications

- Multiple, incremental options
  - No top level justifications (Doug Pearson)
  - Thin Justifications
  - Single thin justification
  - Optimize Top Level Results
- Related options
  - Allow I supported subgoal results
  - Warn if result is I supported

Backtracing / GDS

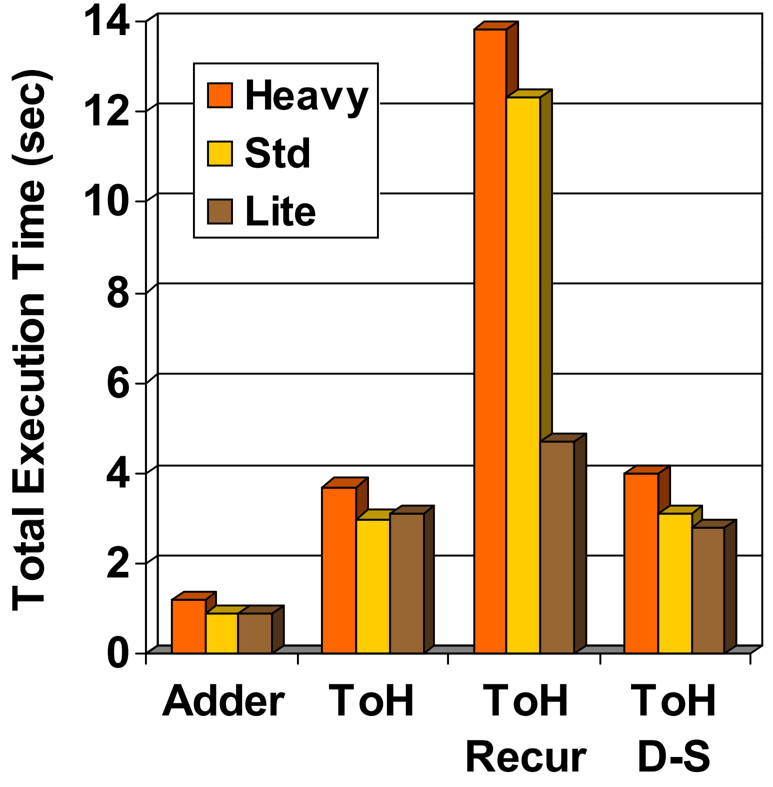
- An option to turn off backtracing
  - "Fake" justifications are grounded using fabricated conditions
- Results:
  - GDS cannot be calculated
  - Learning must also be removed

Using Soar-Lite*



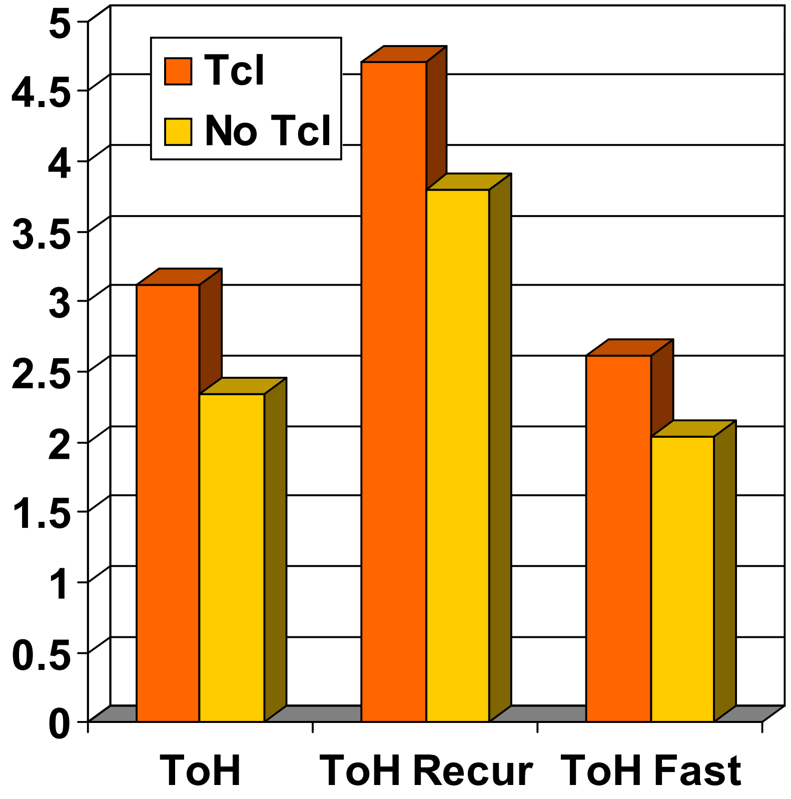
Performance Savings

- 10% savings or better without removing learning
- 15% or better using Soar-Lite
- Up to a *factor of 3* savings using Soar-Lite



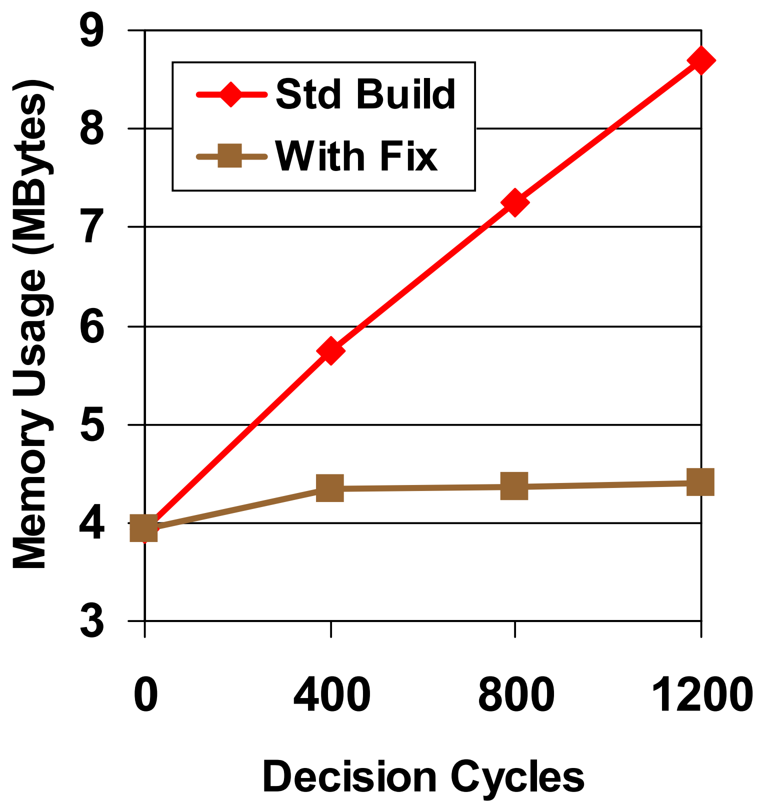
A Shred More Speed

- Some performance gain (~20%) by removing Tcl
- Why?
  - Some overhead from using the GUI ?
  - Initial overhead of calling shared library functions ?



Memory Savings

- Significantly slows memory leak
- Agents which modify structures on the top state benefit most



Nuggets and Coal

- Nuggets
  - Significant speed increases are now ‘easily’ achievable
  - Users have the ability to choose which features are included
  - Some memory issues have been fixed
- Coal
  - Justifications are fairly expensive
  - Soar still can’t count to 2,276,001*