Action Model Learning with Inductive Logic Programming

John Hawkins

Advisor: John Laird

Action Modeling

- Goal is to understand regularities in the environment and the effects of actions
- Such knowledge is difficult to engineer manually (c.f. John Laird's efforts in Tanksoar)
- In the simplistic Tanksoar domain, there are upwards of 2^{80} observable states, even ignoring the limitations of the agents sensors

Why is action modeling hard?

- Actions are complicated/ time consuming to explicate
- Subtleties may make accurate model writing difficult or impossible
- Actions may *change* over time (broken tread example)

Related Work

- Gil, Wang (OBSERVER, 1994), Benson (TRAIL, 1995), and Oates & Cohen have done previous related work
- These efforts each assume a more modest state space, assume that effects of an action are uniquely determined given it's pre-conditions, or assume independence of the features of the state space.

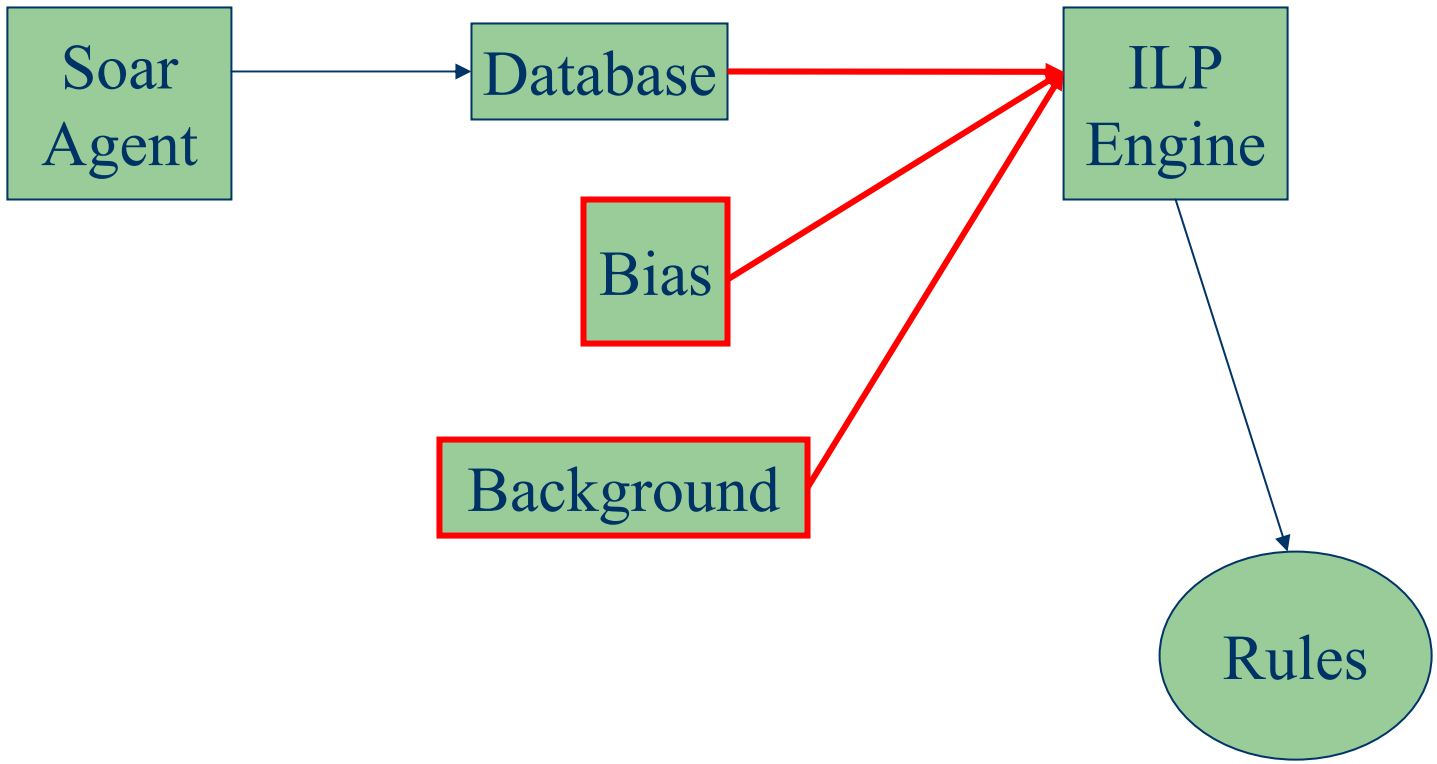
Learning Task

- Sensors & Actions are completely reliable
- Time and Space are discrete
- Environment is deterministic
- Actions have a constant effect horizon
- Each agent controls a single tank
- Examples are correctly labeled

Approach

- Our system used ILP (specifically inverse entailment through the Aleph engine) to learn rules describing the triggering conditions (including actions) for a given predicate
- Experiments focused on the ‘blocked’ sensor for a Tanksoar agent

The Big Picture



ILP

- The system extracts from a behavior trace positive/negative examples of situations that lead to the target predicate value
- An example consists of all sensor predicates from the previous time step and a record of actions taken
- ILP engine produces rules summarizing the examples as generally and accurately as possible.

Summary of Results

- Attempted to learn rules about ‘blocked’ sensor, such as:
  - Turning left while blocked right will result in blocked aft
  - Moving forward with an obstacle ahead and to the left on radar will result in being blocked left
- Results were ultimately unsatisfactory
- Problems can mainly be traced to three key assumptions

Bias Toward Shorter Rules

- Occam's Razor
- If two rules predict equally, choose shorter
- Not always true, due in part to length of structured predicates
- Changing relative size of two predicates cannot fix problem – just leads to mirror difficulties
- But bias necessary to speed search

Quantitative Measures of Quality

- A rule's utility is accurately captured by it's coverage of examples
- Induction process must account for 'noise' in the examples – extremely infrequent cases are nearly impossible to accurately explain
- This leads to ignoring such cases outright
- Could be practical to include background knowledge associating sensors with target predicates they would be likely to affect?

Single Agent Learning

- Assumed system could learn useful information from observing a single agent
  - Necessary both for online action modeling (broken tread) and because only narrow slices of the space result in interesting outcomes. Random behavior will not find these slices frequently.
- Led to rules describing regularities in *agent*
- Unclear that even dropping this assumption would solve problem, as patterns might exist across engineered agents

Nuggets & Coal

- Nuggets
  - Identified important potholes for further research in this area
- Coal
  - Identified them by hitting them