Instance Based Model Learning

Joseph Xu

University of Michigan

Soar Workshop 2009
# Motivation

## 2 extremes of AI problem solving

<table>
<thead>
<tr>
<th>Agent simulates domain interactions in its head, plans fully before acting</th>
<th>Agent has no information about domain, does local search</th>
</tr>
</thead>
<tbody>
<tr>
<td>A* search</td>
<td>Watson Q-Learning</td>
</tr>
<tr>
<td>Engineer must preprogram all aspects of domain into agent</td>
<td>No domain knowledge required</td>
</tr>
<tr>
<td>Agent doesn’t need any experience with the world</td>
<td>Agent needs a lot of experience with the world</td>
</tr>
</tbody>
</table>

### Bridge the gap

- Let the agent learn an (imperfect) domain model from experience with the world
- Agent requires less experience with the world because it generalizes experiences to new situations
3 Basic Research Questions

1. How to generalize experiences into predictive domain models?
2. How to use potentially imperfect domain models to speed up problem solving?
3. How to do all this in the context of Soar?
An Example of Integrated Planning, Learning, and Acting

1. Agent initially has no model of the world, so it just wanders.
2. Agent learns specific model.
3. Agent generalizes plan, which can lead to over-optimistic path.
4. Agent detects mismatch between model and world.
5. New experience refines model.
6. Agent replans.
System Overview

**Planning**
Determine best next action using internal action model.

**Action**
Act in environment.

**Model**
Use experiences to refine internal action model.

- decisions
- predictions
- Experiences
Assumptions About the World

• Deterministic, discrete time steps
• Effects of actions take place in exactly one time step
• Relational representation
  – Only entities are objects, object attributes, and relations on objects
  – Consistent with Soar conventions

(object A) (object B) (object C) (on A B) (ontable B) (ontable C)
Instance-Based Models

**Basic idea**
Predict the outcome of an action (state transition) by making an analogy to a previous episode where the action was performed in a similar state

- Needed: memory of the results of previous actions and ability to search for similar past states
  - Episodic memory naturally fits
- The model is the sum total of all previously experienced state transitions
  - Incremental, one-shot learning
  - More experiences means closer analogies, more likely to be correct
  - Will always converge to perfect accuracy
Episodic Memory Based Models

What happens if I do move(A, C)?

When did I see something similar? (Retrieval)

How are these similar? (Analogical Mapping)

Get a prediction

How does this translate into the state?

What happened next? (Retrieval)
Learning Good Cues

• Epmem will try to match cue as much as possible
• Naïve approach is to use entire current state as cue
• State will contain many features that don’t play a part in determining action effects
• If these distracters are included in the cue, the retrieved state might not be similar in terms of the relevant features

Answer:
Learn to exclude distracters from cue with reinforcement learning

Thanks Nick
Learning Relevant State Features

Incorrect Prediction

Correct Prediction
System Overview

**Planning**

Determine best next action using internal action model.

**Action**

Act in environment.

**Model**

Use experiences to refine internal action model.

decisions

predictions

Experiences
Planning with Learned Models

How do we use possibly imperfect models?

- Problem space search
  - Open-loop policies are vulnerable to single wrong predictions
  - Partial look-ahead is worthless
- Combine look-ahead search with RL
  - Regular Q-learning
  - Use model for shallow look-aheads
  - Back up Q-values
  - Closed loop policies are robust to wrong predictions
RL with 2 Step Look-ahead
RL with 2 Step Look-ahead
RL with 2 Step Look-ahead
Considerations

How do we trade off number of real actions and imaginary updates?

– Look-ahead branching factor, depth

Bad models lead to bad back-ups

– Agent should hold off on look-ahead until it has some confidence in model accuracy

– How to define confidence in model accuracy?
Experimental Setup

Knowledge about when actions can be performed (affordances)

PDDL Domain Specification

Soar Agent
- Action Proposal Rules
- General Procedural Knowledge
- Model
- Control Rules

Domain mechanics

Environment

Episodic Storage

RL
Preliminary Results

4x4 Maze with 7 Step Solution

Is model generalization important?

What if we just did backups over what we’ve already seen?

<table>
<thead>
<tr>
<th>Model Type</th>
<th>Actions to Solve</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plain RL</td>
<td>1000</td>
</tr>
<tr>
<td>EpMem Model</td>
<td>8</td>
</tr>
<tr>
<td>Accum. Exp.</td>
<td>10</td>
</tr>
<tr>
<td>EpMem Model Pre. Exp.</td>
<td>100</td>
</tr>
<tr>
<td>Perfect Model</td>
<td>10</td>
</tr>
</tbody>
</table>

Dyna, 8 updates/step

Plain RL

EpMem Model Accum. Exp.

EpMem Model Pre. Exp.

Perfect Model
Preliminary Results

4 Block World with 6 Step Solution

- Dyna, 8 updates/step
- Plain RL
- EpMem Model

Actions to Solve
Future Work

• Chunk over episodic retrievals to get procedural domain knowledge

• Consider other sources of knowledge when doing prediction
  – Domain independent semantic knowledge such as naïve physics models, object category information
    • YJ’s work – learning semantic categories

• Harder domains – Rogue?
Golden Nuggets
- Model learning is incremental
- Models are guaranteed to converge to perfection
- Can handle any relational domain

Chunks of Coal
- Many algorithms are slow
- Analogical mapping algorithm is naïve
- Results are from trivial problems