RePReL
Integrating Relational Planning and Reinforcement Learning for Effective Abstraction
Overview

**Goal:**
Learning to act in relational domains with varying number of tasks and interacting objects

**Proposed Solution:**
Use a (relational) hierarchical planner to provide abstractions for the RL agents
Toy Example

- Extended Taxi domain
- Planner provides sequence of passenger pickup and drop
Toy Example

- Extended Taxi domain
- Planner provides sequence of passenger pickup and drop
- Learn driving route
RePReL

- Plan the sequence of operators at high level and learn to execute each operator at lower level
- Advantage:
  - Compositionality
  - Task specific abstract representations
- Relational MDP which is deterministic and fully observable
- Adapt First Order Conditional Influence statements to specify bisimulation conditions of MDPs for ‘safe’ abstractions.
Abstraction

First Order Conditional Influence (FOCI) statements

if condition then X1 influence X2

Dynamic FOCI statements

operator : $X_1 \xrightarrow{+1} X_2$

Natarajan, Tadepalli, Dietterich, and Fern 2008
Abstraction

**Given**

State: \{at(p_1, r), taxi-at(13), dest(p_1, y), \neg at-dest(p_1), \neg in-taxi(p_1), at(p_2, b), dest(p_2, g), \neg at-dest(p_2), \neg in-taxi(p_1)\}

Subtask: \langle \text{pickup}(P), \{P/p_1, L/r\} \rangle

D-FOCI:
- \{\text{taxi-at}(L_1), \text{move}(\text{Dir})\} \xrightarrow{+1} \text{taxi-at}(L_2)
- \{\text{taxi-at}(L_1), \text{move}(\text{Dir})\} \xrightarrow{} R

pickup(P):
- \{\text{taxi-at}(L_1), \text{at}(P, L), \text{in-taxi}(P)\} \xrightarrow{+1} \text{in-taxi}(P)
- pickup(P): \text{in-taxi}(P) \xrightarrow{} R_o

**Get**

Abstract state: \{at(p_1, r), taxi-at(13), \neg in-taxi(p_1), \text{move}(\text{Dir})\}
**Abstraction**

**Given**

State: \{at(p1, r), taxi-at(13), dest(p1, y), \neg at-dest(p1), \neg in-taxi(p1),
            at(p2, b), dest(p2, g), \neg at-dest(p2), \neg in-taxi(p1)\}

Subtask: \langle pickup(P), \{P/p1, L/r\} \rangle

D-FOCI:
\{taxi-at(L1), move(Dir)\} \xrightarrow{+1} taxi-at(L2)
\{taxi-at(L1), move(Dir)\} \rightarrow R

pickup(P):
\{taxi-at(L1), at(P, L), in-taxi(P)\} \xrightarrow{+1} in-taxi(P)

pickup(P): in-taxi(P) \rightarrow R_o

**Get**

Abstract state: \{at(p1, r), taxi-at(13), \neg in-taxi(p1), move(Dir)\}

---

 safemodel-agnostic abstraction (Theorem 1)
RePReL Learning Algorithm

- Get high level plan
- For each sub-task
  - Get resp. policy $\pi$
  - Loop till the sub-task is achieved
    - Get the abstract state $s$
    - Get action $a$ from the policy $\pi$
    - Step in env observe reward $\langle s, a, r, s' \rangle$
    - Update the policy $\pi$

*Q learning*
Experiments

- Evaluate for
  - Sample efficiency
  - Transfer across task
  - Generalization across objects
Experiments

- Sample efficiency
- Transfer across task
- Generalization across objects
## Experiments

<table>
<thead>
<tr>
<th></th>
<th>state</th>
<th>abstract state (pick-up)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Relational</strong></td>
<td>at(p1, r), taxi-at(13), dest(p1, y), ¬ at-dest(p1), ¬ in-taxi(p1),</td>
<td>{at(p1, r), taxi-at(13), ¬ in-taxi(p1), move(Dir)}</td>
</tr>
<tr>
<td>[s]</td>
<td>at(p2, b), dest(p2, g), ¬ at-dest(p2), ¬ in-taxi(p1)}</td>
<td></td>
</tr>
<tr>
<td><strong>Vector</strong></td>
<td>[at-p1, dest-p1, in-taxi-p1, at-dest-p2, at-p2, dest-p2,</td>
<td>[at, taxi-at, in-taxi, move ]</td>
</tr>
<tr>
<td>[s,a]</td>
<td>in-taxi-p2, at-dest-p2, taxi-at, move ]</td>
<td></td>
</tr>
</tbody>
</table>
Experiments

Symbol | Meaning
-------|---------
△      | Agent
*      | Furniture
💧     | Coffee machine
✉️     | Mail room
การทำงาน   | Office
A, B, C, D Marked locations

Office world

Craft world

Box world
For human-level general intelligence, the ability to detect compositional structure in the domain and form task-specific abstractions are necessary.
THANKS