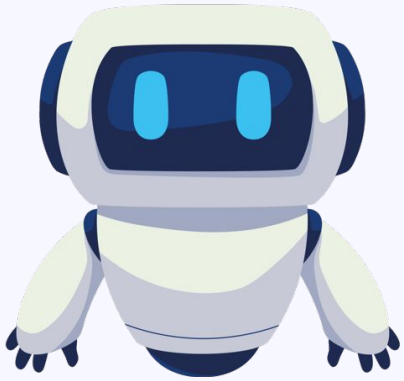
Adding Interactive Task Learning to Thor-Soar

Aaron Mininger

May 5, 2025

45th Soar Workshop

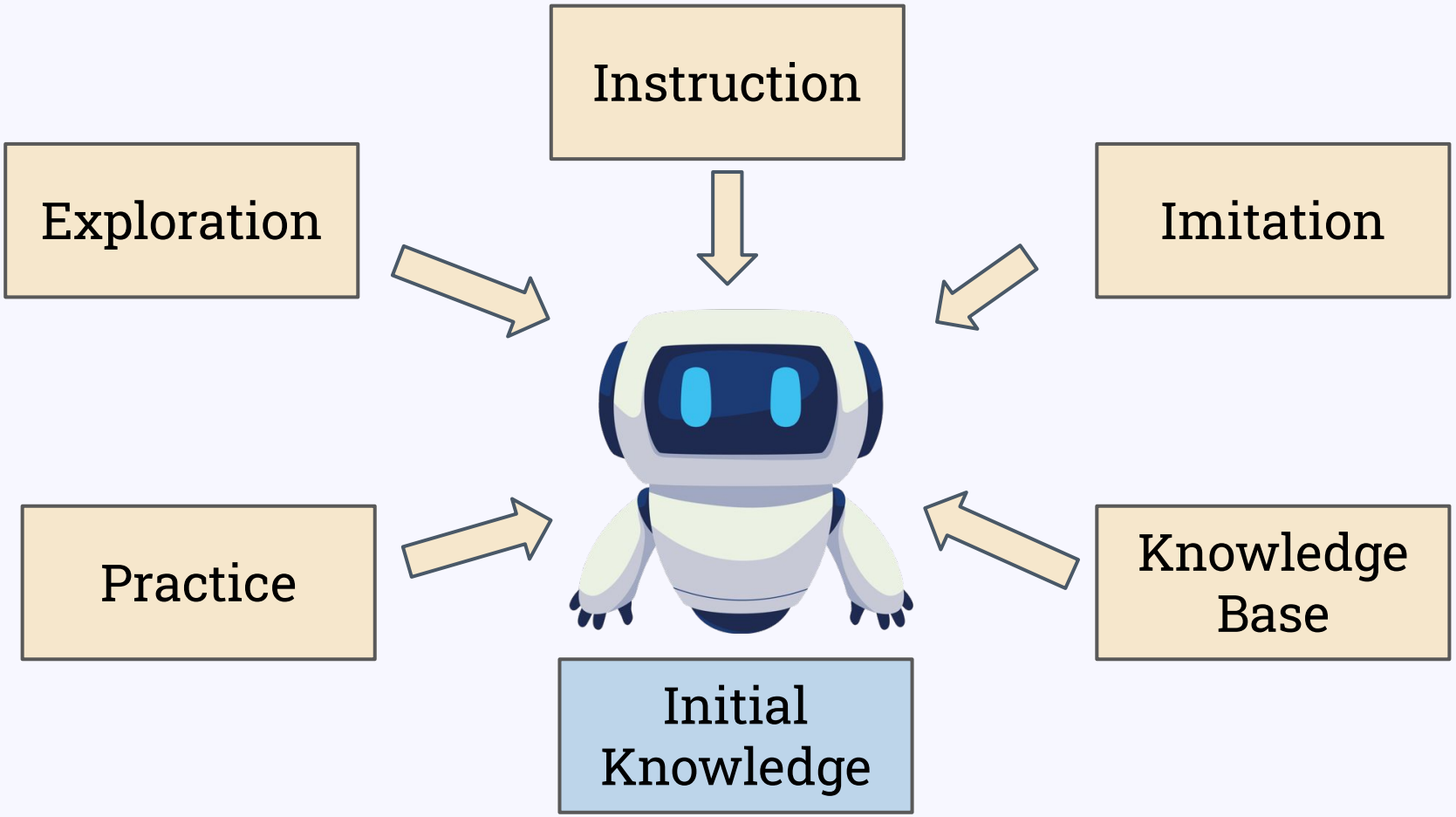
Gaining Task Knowledge



Initial Knowledge

Gaining Task Knowledge

Additional Knowledge Sources



Human Task Learning

Instruction is a highly effective learning source

- High-quality knowledge
- Contains feedback + corrections
- Leverages domain knowledge
- Data efficient (rapid learning)

Interactive Task Learning

Design agents that can learn new tasks from scratch through natural forms of interaction

Situated Interactive Instruction

Situated

Instruction happens in a shared environment

Interactive

Both the instructor and agent engage in dialog

Instruction

Agent learns primarily through natural language

Learning Problem

Key Characteristic: The agent must learn quickly from few examples

Learning must be:

- **Efficient:** Maximize learning from each instruction
- **Generalizable:** Apply learning to future task variations
- **Compositional:** Build on previously learned knowledge
- **Diverse:** Learn a range of task and knowledge types

Thor Soar

Thor Soar Domain

Kitchen domain using ai2thor

*ai2thor.allenai.org

Thor Soar Domain

Initial Knowledge

- World Management
- Object Ontology/Affordances
- NL Comprehension
- Primitive Actions (open, turn-on)
- Planning via Means-Ends Analysis

Simple Task Example

Pick up the apple.

Step 1: Comprehension

Pick up the apple.

Comprehension

Message Structure

```
(<msg> ^class command
       ^action <action>
       ^argument1 <arg1>)
  (<action> ^action-handle pick-up
            ^goal <goal>)
  (<arg1> ^instance-of reference-description
          ^grounding <grounding1>)
    (<grounding1> ^category apple
                  ^object-handle Apple1)
```

Step 2: Interpretation

Pick up the apple.

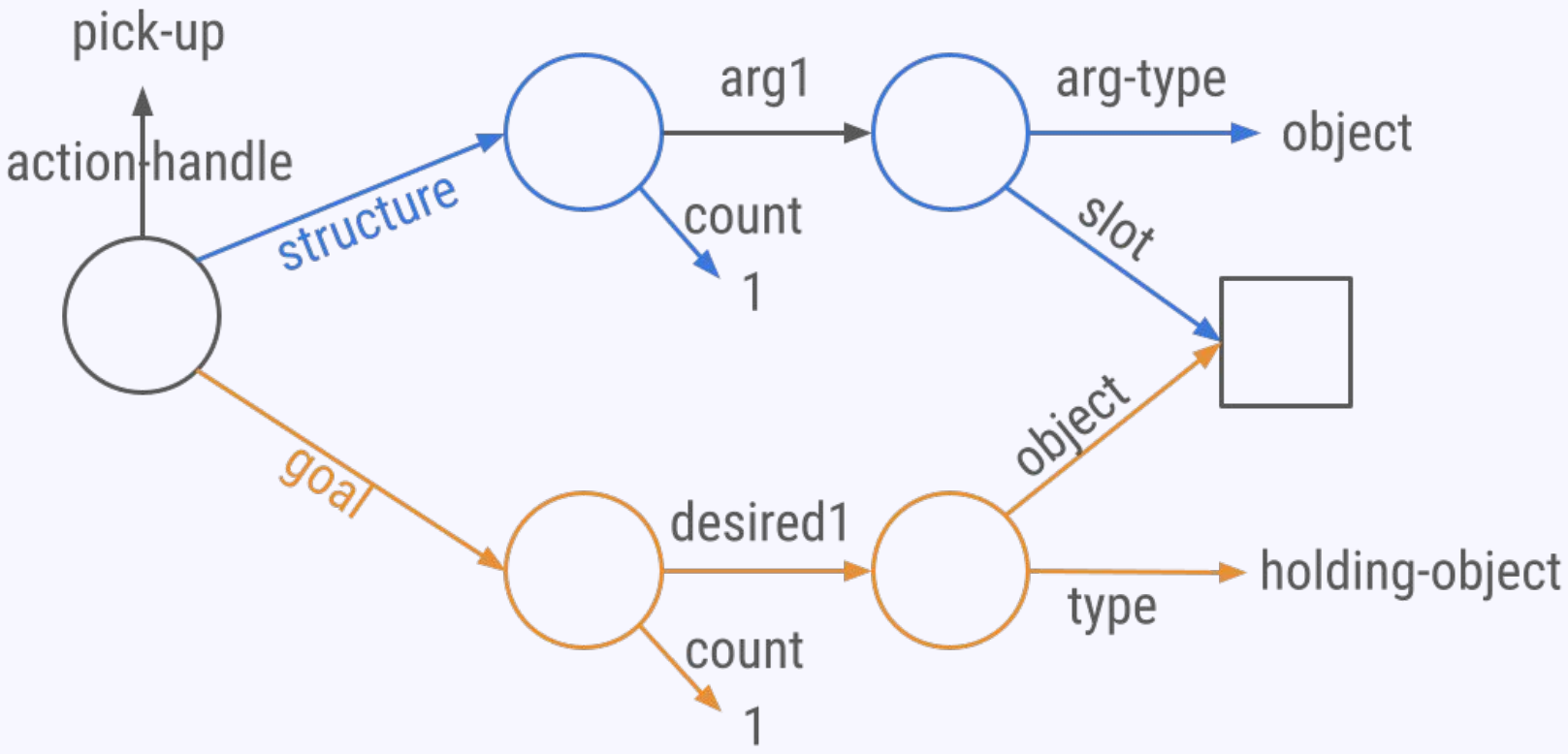
Comprehension

Message Structure

```
(class=command, action=pick-up,
   arg1=Apple1)
```

Action Network

Action network defined in semantic memory:



Step 2: Interpretation

Pick up the apple.

Comprehension

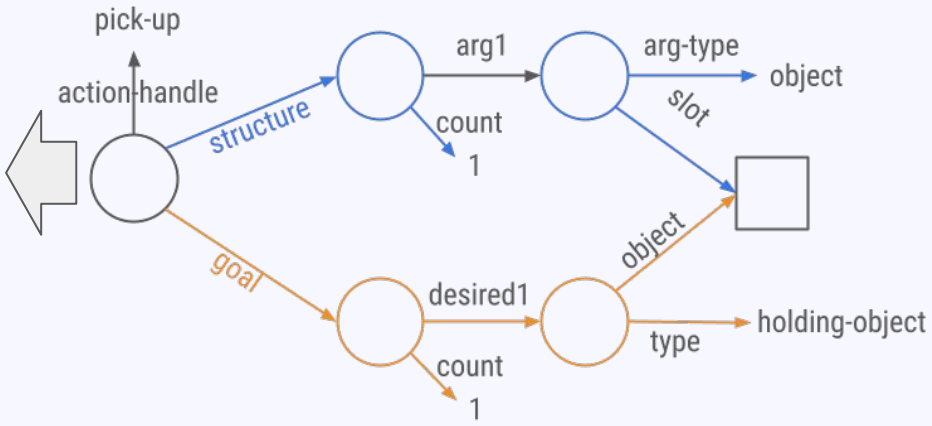
Message Structure

```
(class=command, action=pick-up,
   arg1=Apple1)
```

Interpretation

Desired (goal)

```
desired = holding-object(Apple1)
```



Step 3: Execution

Pick up the apple.

Comprehension

Message Structure

```
(class=command, action=pick-up,
   arg1=Apple1)
```

Interpretation

Desired (goal)

```
desired = holding-object(Apple1)
```

Planning + Execution

```
execute: approach(Apple1)
```

```
execute: pick-up(Apple1)
```

Step 3: Execution

Pick up the apple.

Comprehension

Message Structure

```
(class=command, action=pick-up,
   arg1=Apple1)
```

Interpretation

Desired (goal)

```
desired = holding-object(Apple1)
```

Planning + Execution

```
execute: approach(Apple1)
```

```
execute: pick-up(Apple1)
```

What happens if the agent does not have this task knowledge?

Task Learning Example

Discard the apple.

Task Learning Example

Discard the apple.

Comprehension

Message Structure

```
(class=command, action=discard,
   arg1=Apple1)
```

Task Learning Example

Discard the apple.

Comprehension

Message Structure

```
(class=command, action=discard,
   arg1=Apple1)
```

Interpretation

Failure! No network in smem

Learning: Discard

What is the goal of discard?

The goal is that the apple is in the garbage can.

Learning: Discard

What is the goal of discard?

The goal is that the apple is in the garbage can.

Comprehension

Goal Structure

```
(type=relation, relation-handle=in,
  object1=Apple1, object2=GarbageCan37)
```

Learning: Discard

Message Structure

```
(class=command, action=discard,
   arg1=Apple1)
```

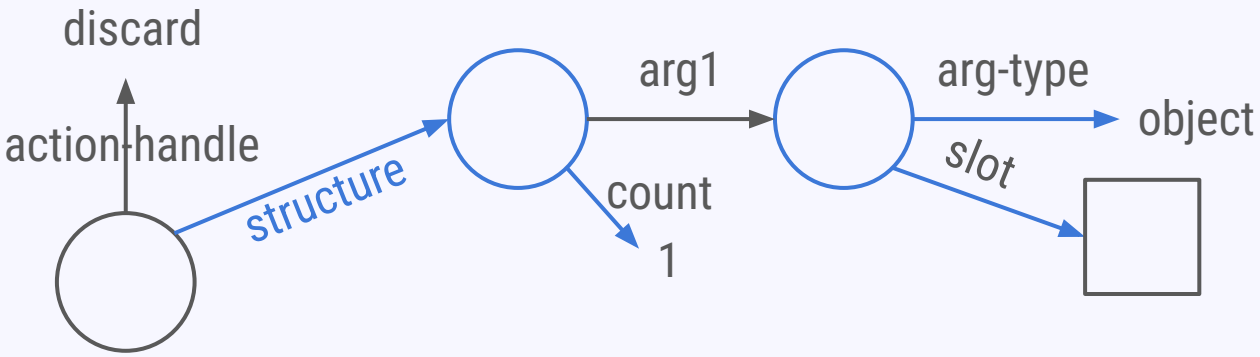
Goal Structure

```
(type=relation, relation-handle=in,
  object1=Apple1, object2=GarbageCan37)
```

Learning: Discard

Message Structure

```
(class=command, action=discard,
   arg1=Apple1)
```



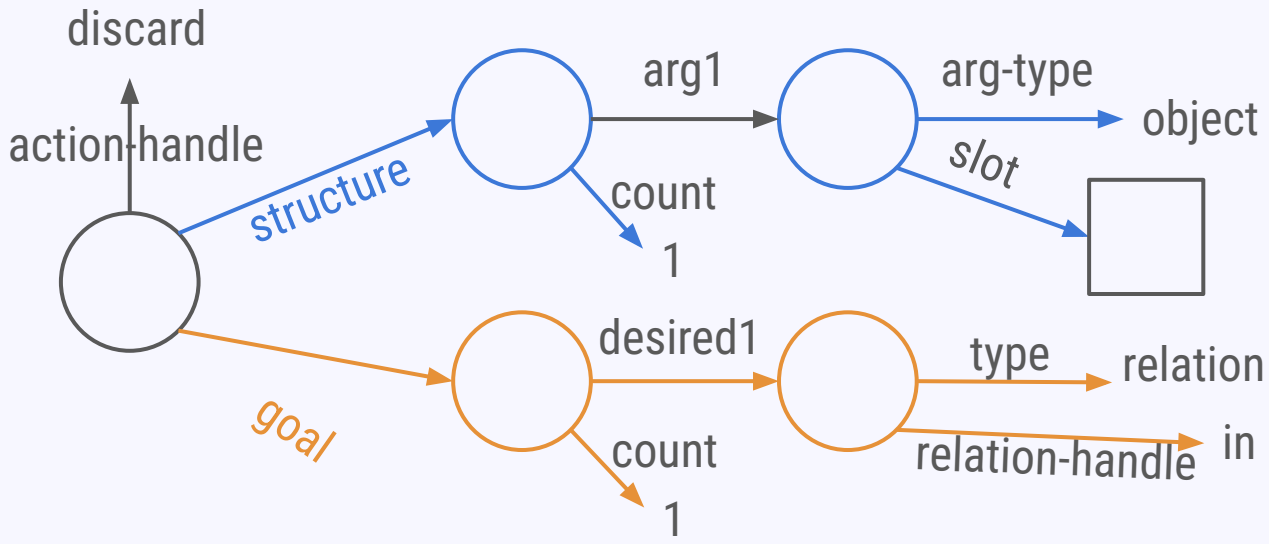
Learning: Discard

Message Structure

```
(class=command, action=discard,
   arg1=Apple1)
```

Goal Structure

```
(type=relation, relation-handle=in,
  object1=Apple1, object2=GarbageCan37)
```



Learning: Discard

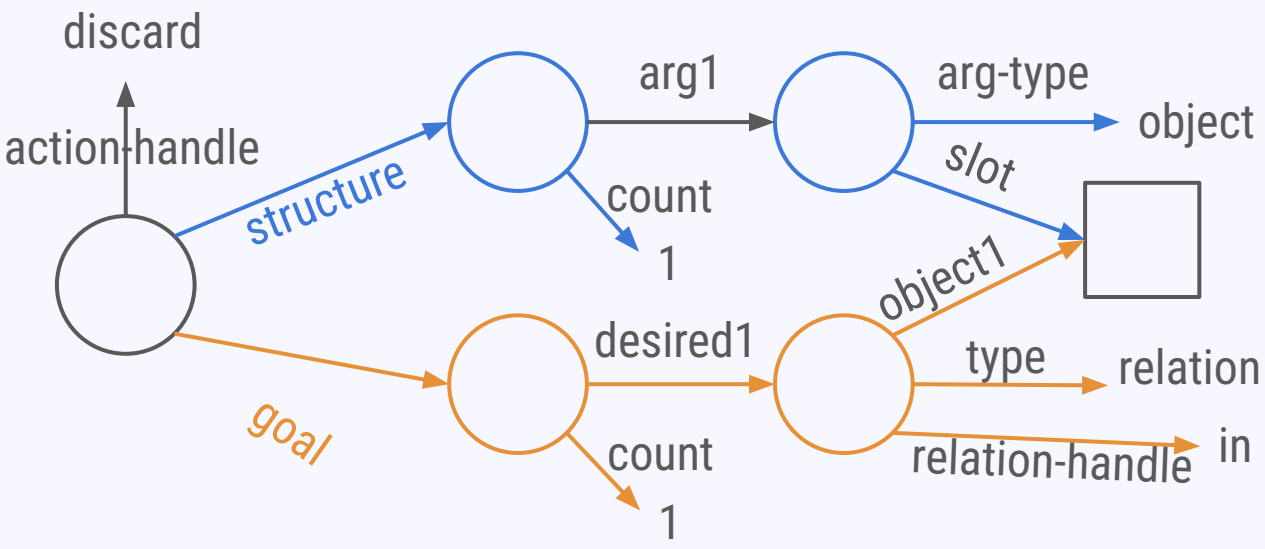
Message Structure

```
(class=command, action=discard,
   arg1=Apple1)
```

Goal Structure

```
(type=relation, relation-handle=in,
  object1=Apple1, object2=GarbageCan37)
```

Same Object → Explicit Argument



Learning: Discard

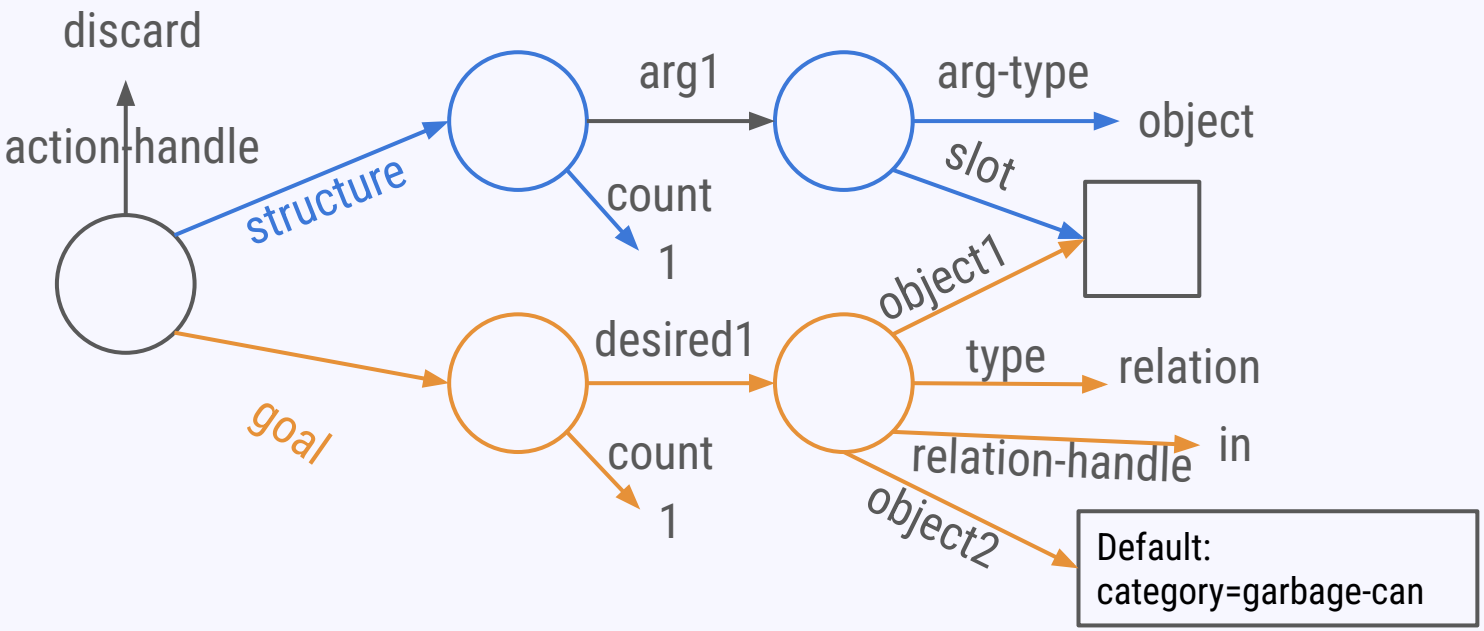
Message Structure

```
(class=command, action=discard,
   arg1=Apple1)
```

Goal Structure

```
(type=relation, relation-handle=in,
  object1=Apple1, object2=GarbageCan37)
```

Only in goal → Implicit Argument



Learning: Discard

Discard the potato.

Learning: Discard

Discard the potato.

Comprehension

Message Structure

```
(class=command, action=discard,
   arg1=Potato6)
```

Interpretation

Desired (goal)

```
desired = in(Potato6, GarbageCan37)
```

Planning + Execution

```
execute: pick-up(Potato6)
```

```
execute: approach(GarbageCan37)
```

```
execute: put-in(Potato6, GarbageCan37)
```

🟡 Nuggets

The agent can learn goals comprised of
1 or more property, relation, or holding desireds

Examples:

- **Shut the fridge:** The goal is that the fridge is closed.
- **Serve the potato:** The goal is that the potato is cooked and the potato is on the plate.
- **Store the egg:** The goal is that the egg is in the fridge and the fridge is closed.

Coal

Limitations:

- **Limited to goal-based tasks**
- **No goal variations:** Only 1 goal per action
- **Can overgeneralize**

Next Steps

Currently, the agent requires a lot of innate knowledge about its basic actions.

Can we learn this?

Next Steps

Currently, the agent requires a lot of innate knowledge about its basic actions.

Can we learn this?

- Preconditions
- Effects + Side Effects
- Preferences + Search Knowledge

Example: open

Action Command: open <obj>

- **Preconditions:** reachable(<obj>), holding(none)
- **Effect:** open(<obj>)
- **Command:** open

Example: open

MEA Proposal Rule

```
if:
   unachieved-desired = open(<obj>)
 then:
   perform command open with <obj>
   preconditions = { reachable(<obj>),
                     holding-object(none) }
```

Questions?