

### abstraction

 negative absent (:action load\_truck :parameters (?package ?truck ?location) :precondition (and (at ?truck ?location) Concrete model (at ?package ?location)) :effect (and (not (at ?package ?location)) (in ?package ?truck)))

# Algorithm

- Start with the most abstracted node in lattice.
- Pick abstraction candidates in some order.
- For each candidate, generate three models and for each pair of models:
- Generate a distinguishing query Q and pose it to the agent.
- Get the response R from the agent.
- Prune out the incorrect variants of candidate models.
- Repeat steps 3-6 till the model is fully estimated.
- Return the final set of model(s).

# Learning Interpretable Models for Black-Box Agents Pulkit Verma and Siddharth Srivastava, Arizona State University, Tempe, AZ, USA

### Example of Agent Interrogation

#### Plan Outcome Query: Asks the outcome of a plan.

Query: Initial state, plan. **Response**: Length of successful execution, final state.



can appear in three forms: positive

# Key Algorithmic Principle

Key feature of the algorithm time we prune an Each abstracted model, we prune a very large number of models at the most concrete node.



What do you think will happen if you execute the plan  $\pi$ : (load\_truck (p1, t1), drive (t1, Paris, Pisa)) starting in an initial state s<sub>I</sub>: (at (t1, Paris) ^ at (p1, Paris))?

can execute only the first step, and the final state after executing one step was *s<sub>F</sub>*:(¬at (p1, Paris) ^ in (t1, p1) ^ at(t1, Paris))

(:action load truck								
` :parameters (?package ?truck ?location)								
:precondition (and (+/-/Ø) (at ?truck ?location)	$n_1$							
(+/-/Ø) (in ?package ?truck)	$n_2$							
(+/-/Ø) (at ?package ?location))	$n_3$							
:effect (and (+/-/Ø) (at ?package ?location)	$n_4$							
(+/-/Ø) (at ?truck ?location)	$n_5$							
(+/-/Ø) (in ?package ?truck)))	$n_6$							
<ul> <li>Model estimate</li> <li>before querying</li> </ul>								
Estimated Model(s)								
after querying								
(:action load_truck								
:parameters (?package ?truck ?location)								
:precondition (and (at ?truck ?location)								
(-/Ø) (in ?package ?truck)								
(at ?package ?location))								
:effect (and (not (at ?package ?location))								
(in 2nockore 2truck (Incation)								
(in spackage struck)))								

#### Results

- Randomly generate an agent and environment from the IPC benchmark suite.
- Algorithm learns this agent's model.

Domain	$ \mathbb{P} $		<b>Q</b> naive	$ \mathcal{M}_{equiv} $	$ \mathcal{M} $	<b> 2 </b>	Time/ <b>Q</b> (sec)
gripper	5	3	$15 \times 2^{5}$	1	1	37	0.14
blocks-world	9	4	36 × 2 <sup>9</sup>	1	1	92	1.73
elevator	10	4	$40 \times 2^{10}$	64	1	109	5.91
logistics	11	6	$66 \times 2^{11}$	64	1	98	11.62
parking	18	4	72 × 2 <sup>18</sup>	32	1	173	12.01
satellite	17	5	85 × 2 <sup>17</sup>	4096	1	127	19.53

Number of queries (|Q|) and number of models ( $|\mathcal{M}|$ ) generated in our approach vs naïve baseline. Results are averages of 10 random runs.

# Salient Features

- capabilities.

#### This work was supported in part by the NSF under grants IIS 1844325 and IIS 1909370

![](_page_0_Picture_39.jpeg)

![](_page_0_Picture_40.jpeg)

![](_page_0_Picture_41.jpeg)

**Theorem**: The algorithm will always terminate and return a set of models, each of which are functionally equivalent to agent's model.

• Needs no prior knowledge of the agent model. • Requires an agent to have only rudimentary query answering

• Queries can be answered by the agent using a simulator. • Works for non-stationary environments.

![](_page_0_Picture_46.jpeg)