# Integrated Planning and Reinforcement Learning for Compositional Domains

—

Harsha Kokel
Research Scientist
IBM Research

# Integrated Planning and Reinforcement Learning for Compositional Domains

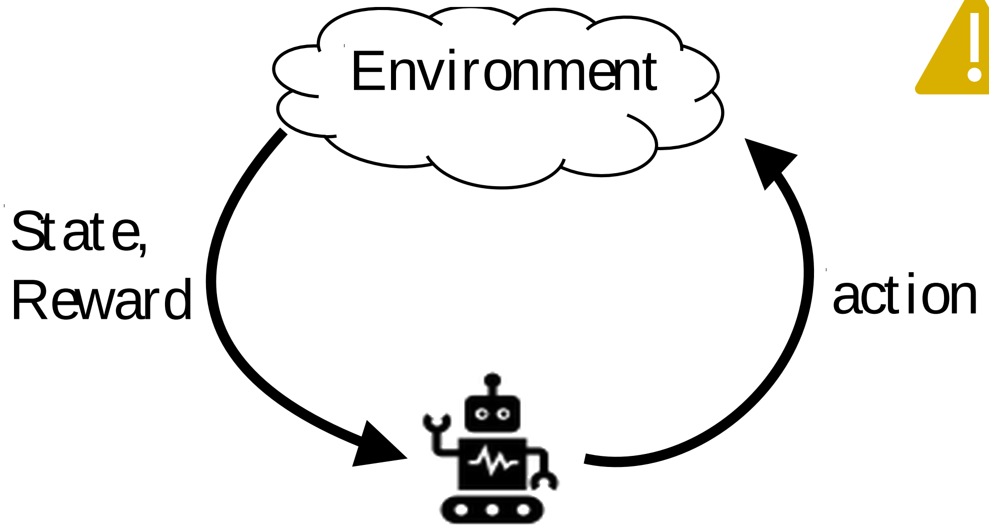
—

Harsha Kokel
Research Scientist
IBM Research

Prof. Sriraam Natarajan (left)

StARLinG LAB

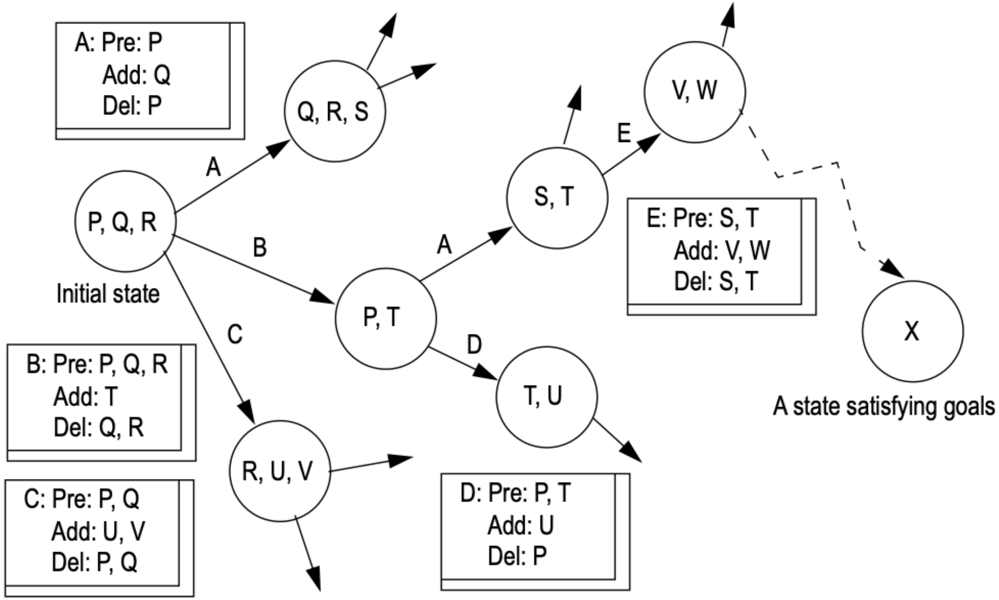
# Reinforcement Learning





Relational MDP

# Planning



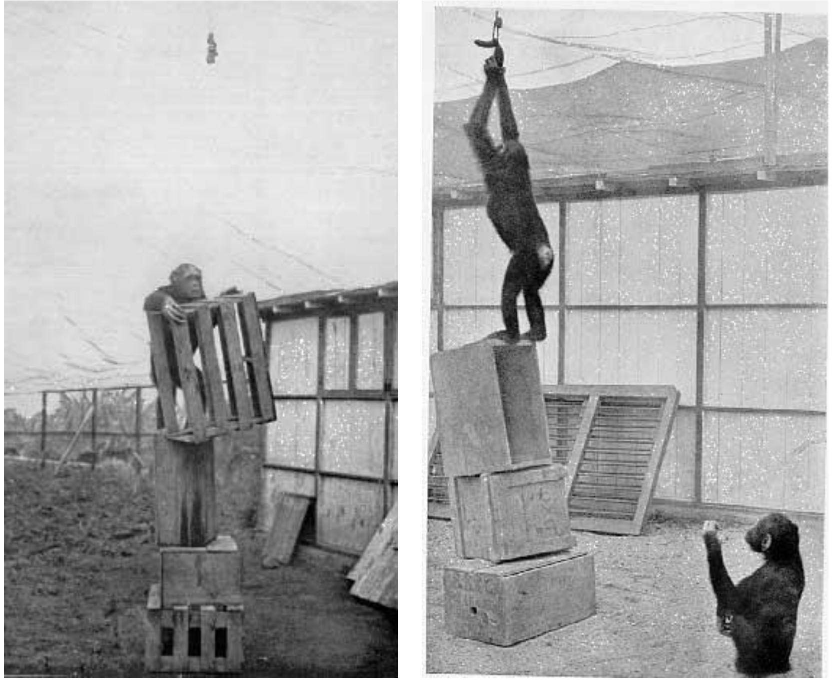

Long and Fox (2002)

## Planning

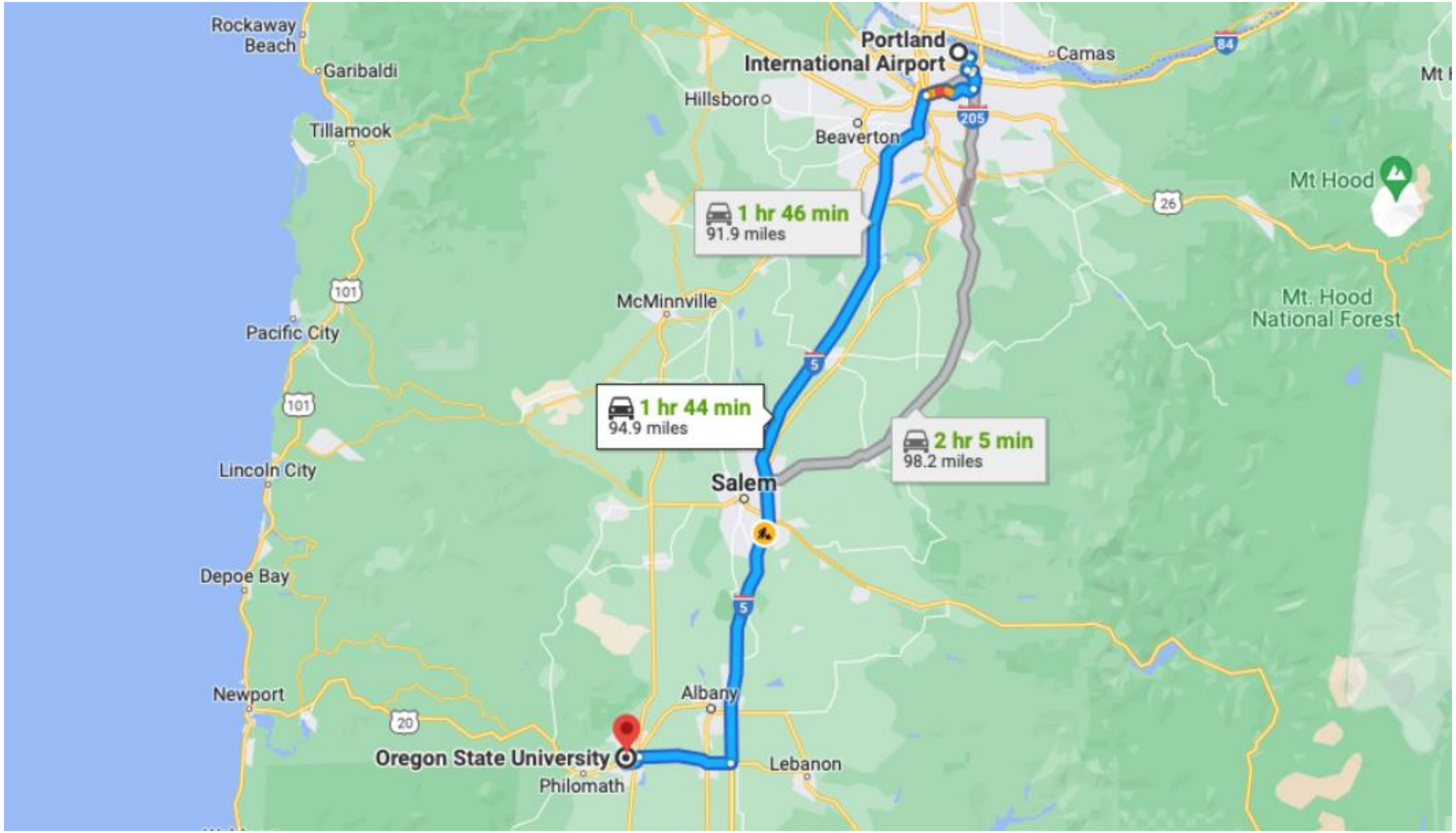
- Search through space of states
- Relies on an explicit model of the environment

## RL

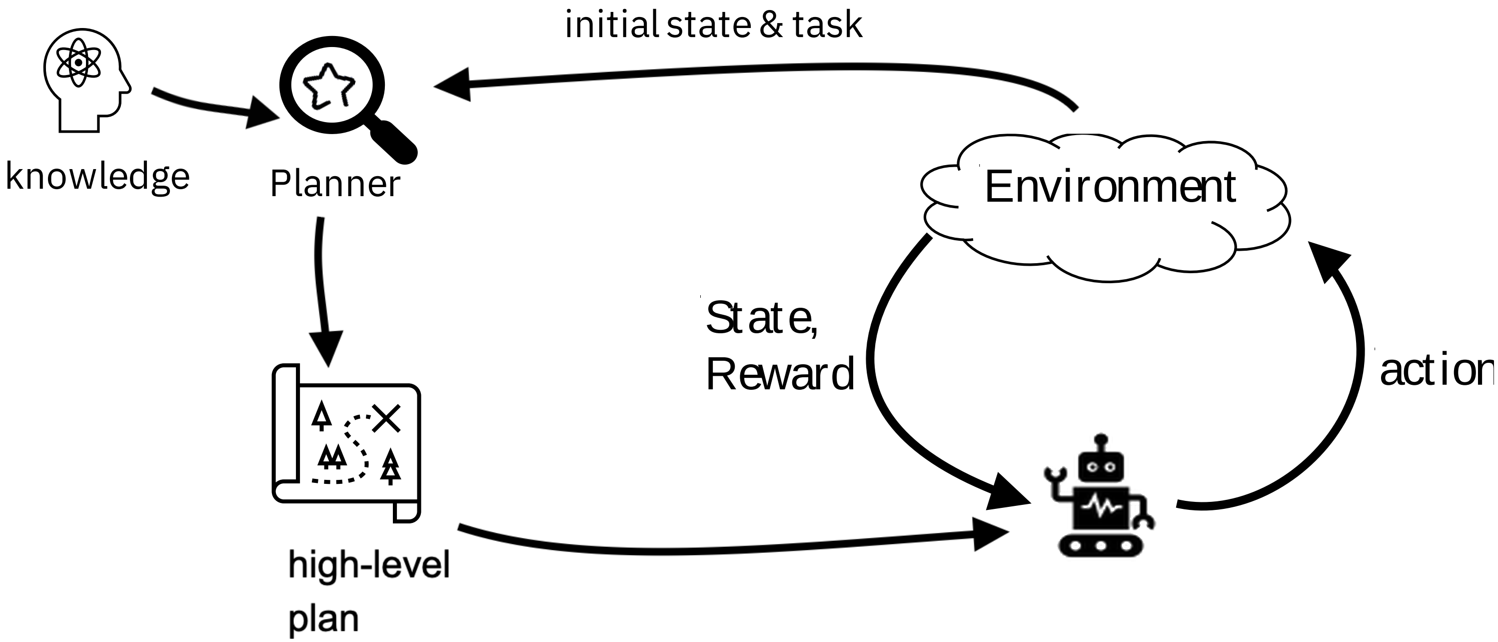
- Search through space of policies
- Relies on trial & error by interaction



Köhler (1948)


Rockaway Beach
Garibaldi
Tillamook
Pacific City
Lincoln City
Depoe Bay
Newport
Portland International Airport
Camas
Hillsboro
Beaverton
205
84
Mt Hood
26
Mt. Hood National Forest
1 hr 46 min
91.9 miles
McMinnville
101
5
1 hr 44 min
94.9 miles
2 hr 5 min
98.2 miles
Salem
Albany
20
Oregon State University
Philomath
Lebanon

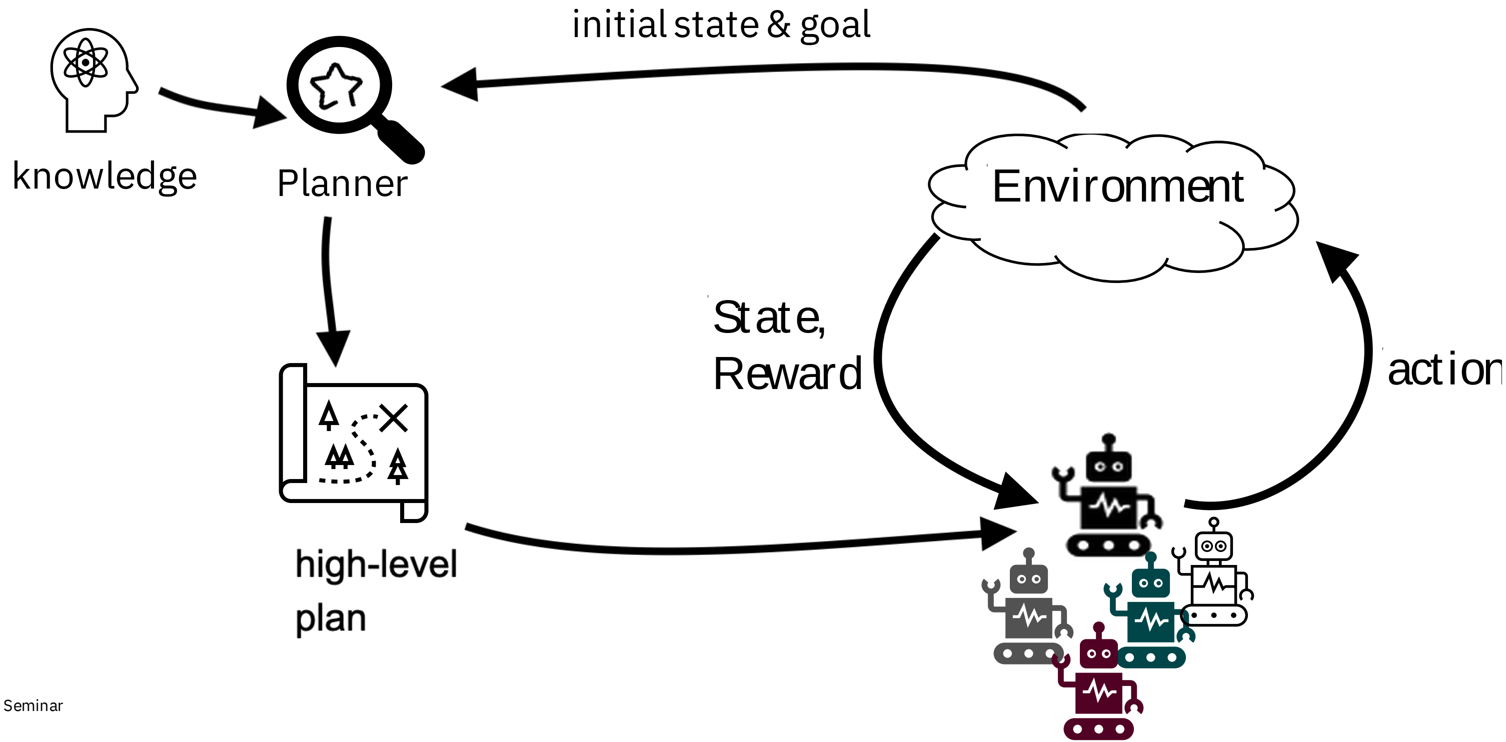
# Integrating Planning and RL




EXIT 288
5 NORTH
Portland
205 NORTH
Oregon City
EXIT
ONLY

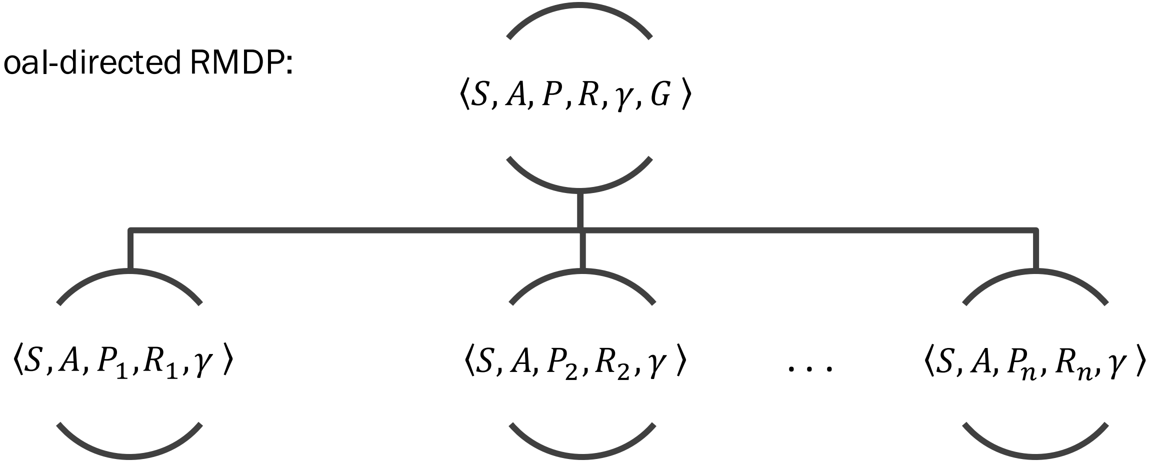

LEFT TURN
YIELD
ON
FLASHING
ARROW

# Integrating Planning and RL



# Decomposing GRMDP

Goal-directed RMDP:



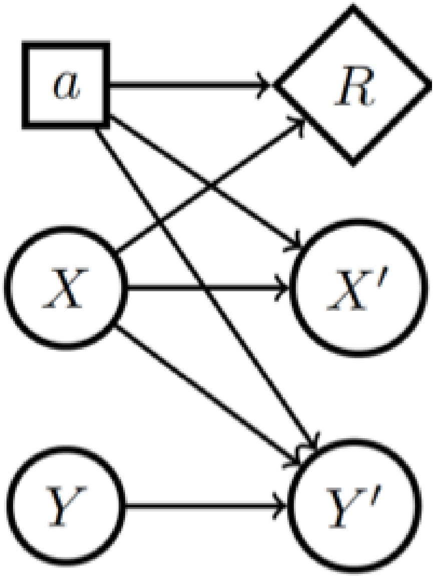

. . .

Multiple subgoal RMDP:

$$R_o(s,a,s') = \begin{cases} t_R + R(s,a,s') & \textit{if } s' \in \beta(o) \textit{ and } s \notin \beta(o) \\ 0 & \textit{if } s' \in \beta(o) \textit{ and } s \in \beta(o) \\ R(s,a,s') & \textit{otherwise} \end{cases}$$

$$P_o(s,a,s') = \begin{cases} 0 & \textit{if } s \in \beta(o) \textit{ and } s' \notin \beta(o) \\ 1 & \textit{if } s \in \beta(o) \textit{ and } s' \in \beta(o) \\ P(s,a,s') & \textit{otherwise} \end{cases}$$

# Irrelevant variables




Factored MDP represented as Dynamic Bayesian Network (DBN)

# Model-agnostic Abstraction

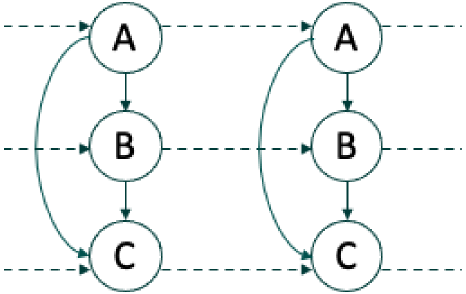
A **model-agnostic abstraction** $\phi(s)$ is such that for any action $a$ and an abstract state $\bar{s}$, $\phi(s_1)$=$\phi(s_2)$ if and only if

$$\sum_{\{s'_1 \mid \phi(s'_1)=\bar{s}\}} R_o(s_1, a, s'_1) = \sum_{\{s'_2 \mid \phi(s'_2)=\bar{s}\}} R_o(s_2, a, s'_2)$$

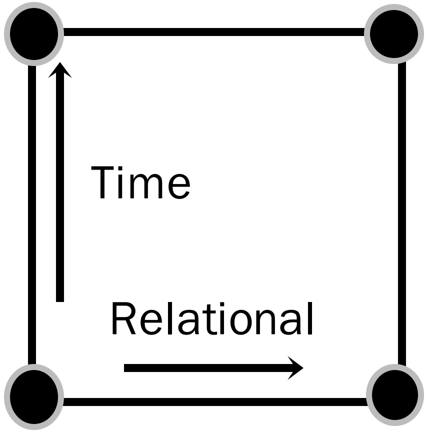
$$\sum_{\{s'_1 \mid \phi(s'_1)=\bar{s}\}} P_o(s_1, a, s'_1) = \sum_{\{s'_2 \mid \phi(s'_2)=\bar{s}\}} P_o(s_2, a, s'_2)$$


Dietterich NeurIPS 2000; Ravindran and Barto IJCAI 2003;
Givan, Dean, and Greig AI 2003; Li, Walsh, and Littman ISAIM 2006

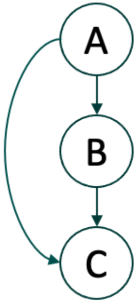
# Graphical models




Dynamic
BN

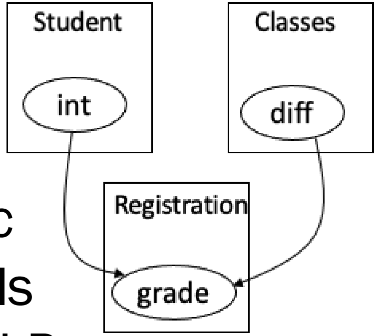



Dynamic
PLM




Bayesian Network
(BN)

Probabilistic
Logic Models
(PLM, PRM, BLP,
LBN, DAPER)




Koller and Friedman 2009;
Getoor and tasker 2007; Raedt et al. 2016

# D-FOCI

First Order Conditional Influence (FOCI) statements

$$\texttt{if}\ \langle condition \rangle \quad \texttt{then}\ \langle influents \rangle \quad \texttt{QINF}\ \langle resultant \rangle$$

Dynamic FOCI statements

$$[\langle subgoal \rangle]\text{:}\ \langle influents \rangle \ \xrightarrow{[+1]} \ \langle resultant \rangle$$

Natarajan, Tadepalli, Dietterich, and Fern 2008

# D-FOCI

$$\{\texttt{action},\texttt{taxi\_at}(X)\} \xrightarrow{+1} \texttt{taxi\_at}(X) \quad \text{(3a)}$$

$$\texttt{pick}(P) : \{\texttt{action}, \texttt{taxi\_at}(X), \texttt{at}(P, Y),$$

$$\texttt{in\_taxi}(P)\} \xrightarrow{+1} \texttt{in\_taxi}(P) \quad \text{(3b)}$$

$$\texttt{pick}(P) : \{\texttt{in\_taxi}(P)\} \longrightarrow Reward \quad \text{(3c)}$$

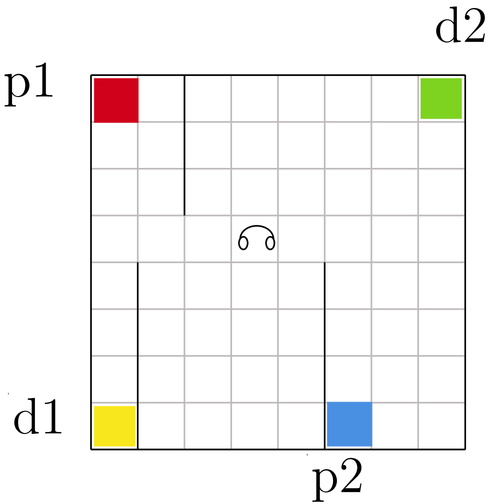
$$\texttt{drop}(P) : \{\texttt{at\_dest}(P)\} \longrightarrow Reward \quad \text{(3d)}$$

$$\texttt{drop}(P) : \{\texttt{at}(P, X), \texttt{dest}(P, D), \texttt{at\_dest}(P)\}$$

$$\longrightarrow \texttt{at\_dest}(P) \quad \text{(3e)}$$

$$\texttt{drop}(P) : \{\texttt{action}, \texttt{taxi\_at}(X), \texttt{at}(P, Y),$$

$$\texttt{in\_taxi}(P)\} \xrightarrow{+1} \texttt{at}(P, K) \quad \text{(3f)}$$



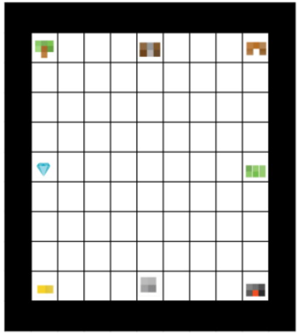
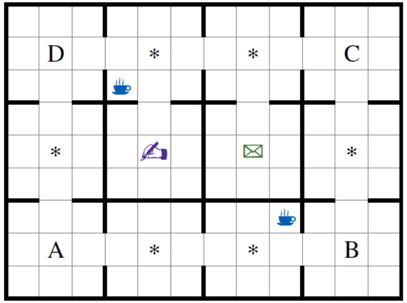
# Experiments

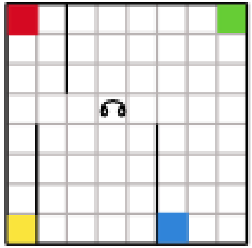
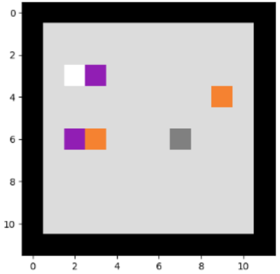
Domains

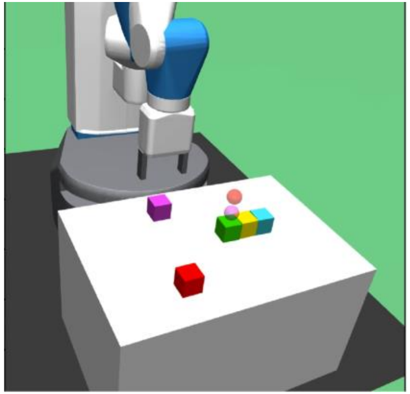
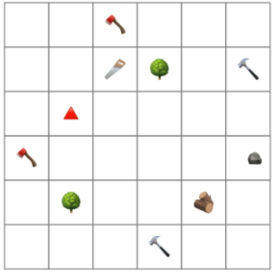
- Office World
- Minecraft World
- Relational Taxi
- Relational Box World
- Craft World
- Robotic Fetch domain

Baselines

- HRL (options framework)
- Tabular Q-learning
- Deep RL (DDQN, HDQN, SAC)
- Deep Relational RL
- Planning+RL (Taskable RL)

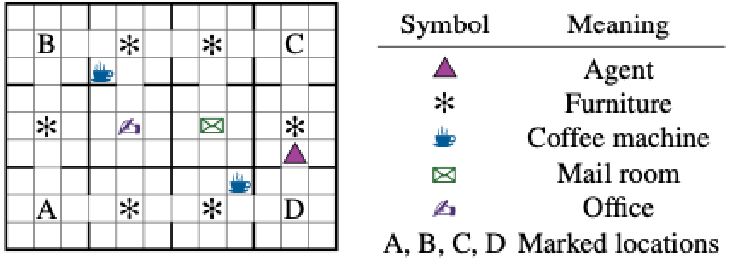



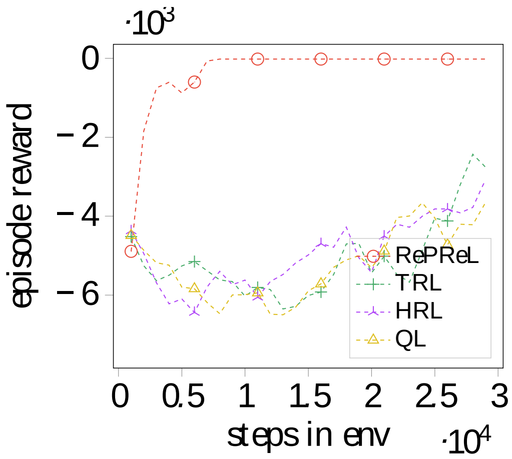
# Experiments

Office World

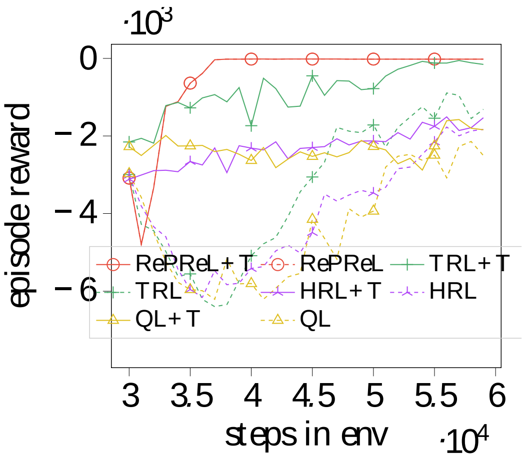



Sample efficiency

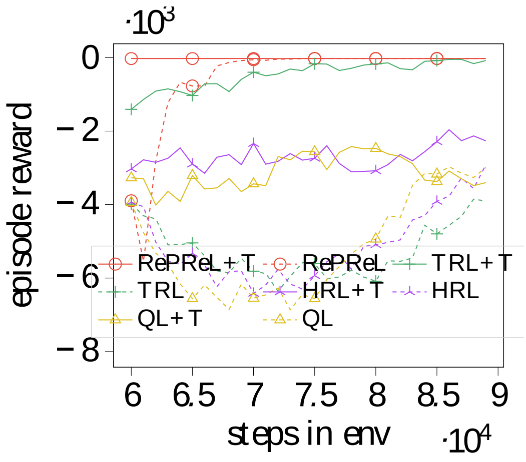
Transfer across task




Deliver mail



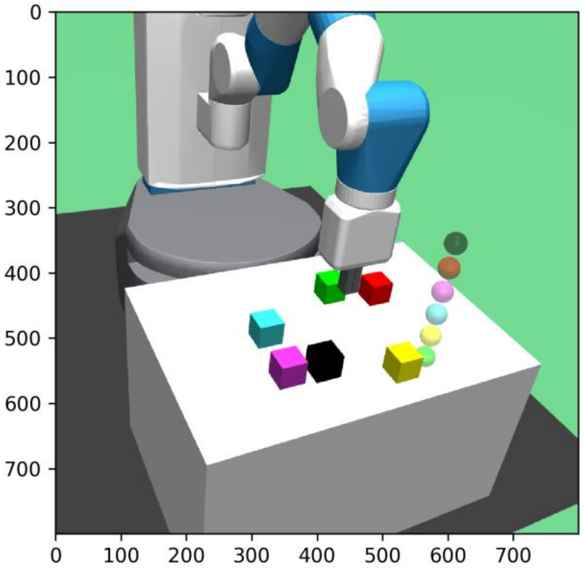

Deliver coffee

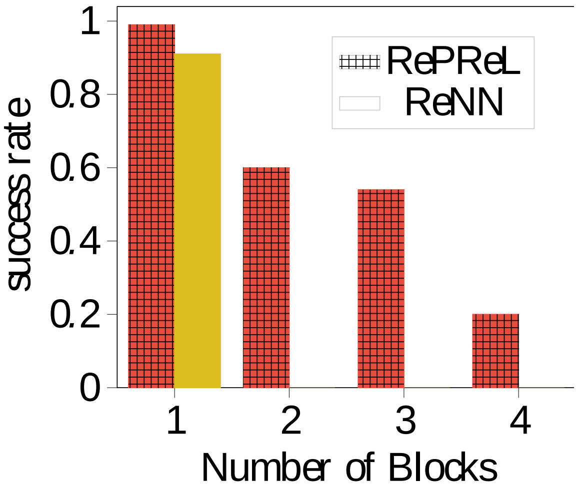


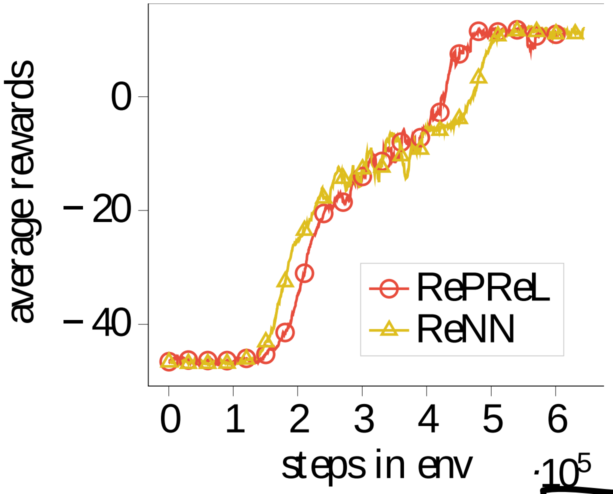

Deliver mail and coffee

Kokel et al. ICAPS 2021

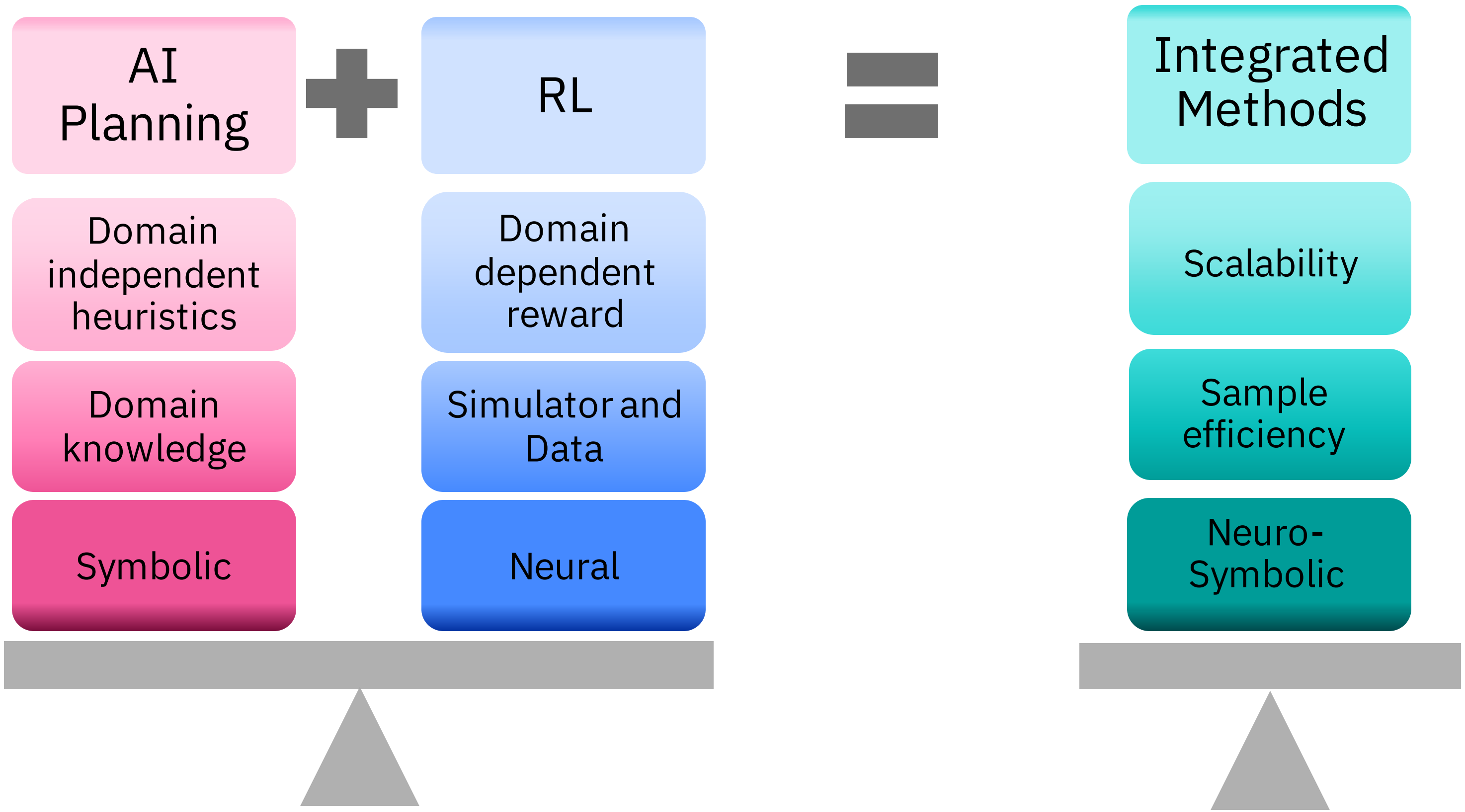
# Deep Relational RL



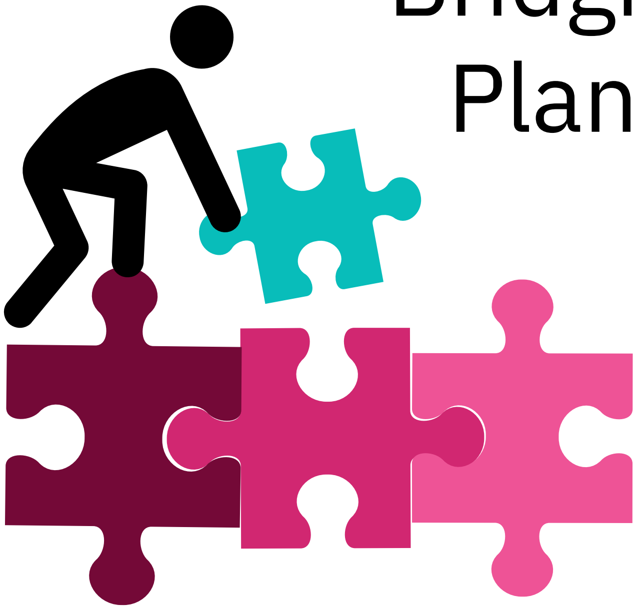



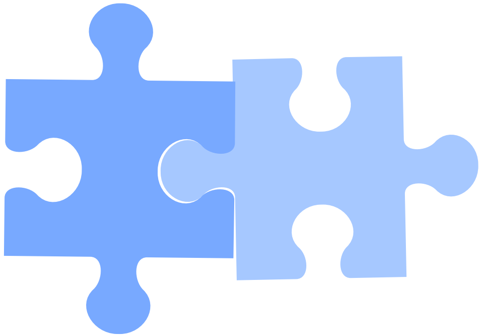




[1]Kokel et al. NCAA 2022a
[2]Li et al. ICRA 2022


AI Planning
Domain independent heuristics
Domain knowledge
Symbolic
RL
Domain dependent reward
Simulator and Data
Neural
Integrated Methods
Scalability
Sample efficiency
Neuro-Symbolic

# Bridging the Gap
# Planning & RL





## Planning

- Search through space of states
- Relies on an explicit model of the environment
- PDDL Task

## RL

- Search through space of policies
- Relies on trial & error by interaction
- MDP

# PDDL Task

$\langle \mathcal{L}, \mathcal{O}, I, G \rangle$

## Lifted Action Models $\mathcal{O}$

```
(:action move
:parameters (?curpos ?nextpos ?dir)
:precondition (and (place ?curpos)
   (place ?nextpos) (at-robot ?curpos)
   (conn ?curpos ?nextpos ?dir) (open ?nextpos))
:effect (and (at-robot ?nextpos)
           (not (at-robot ?curpos))))
```

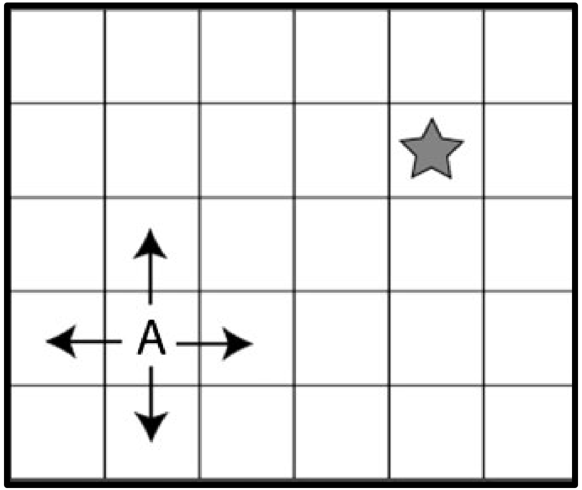
```
move(c_1_1, c_1_2, east),
move(c_2_2, c_2_1, west),    ...
```

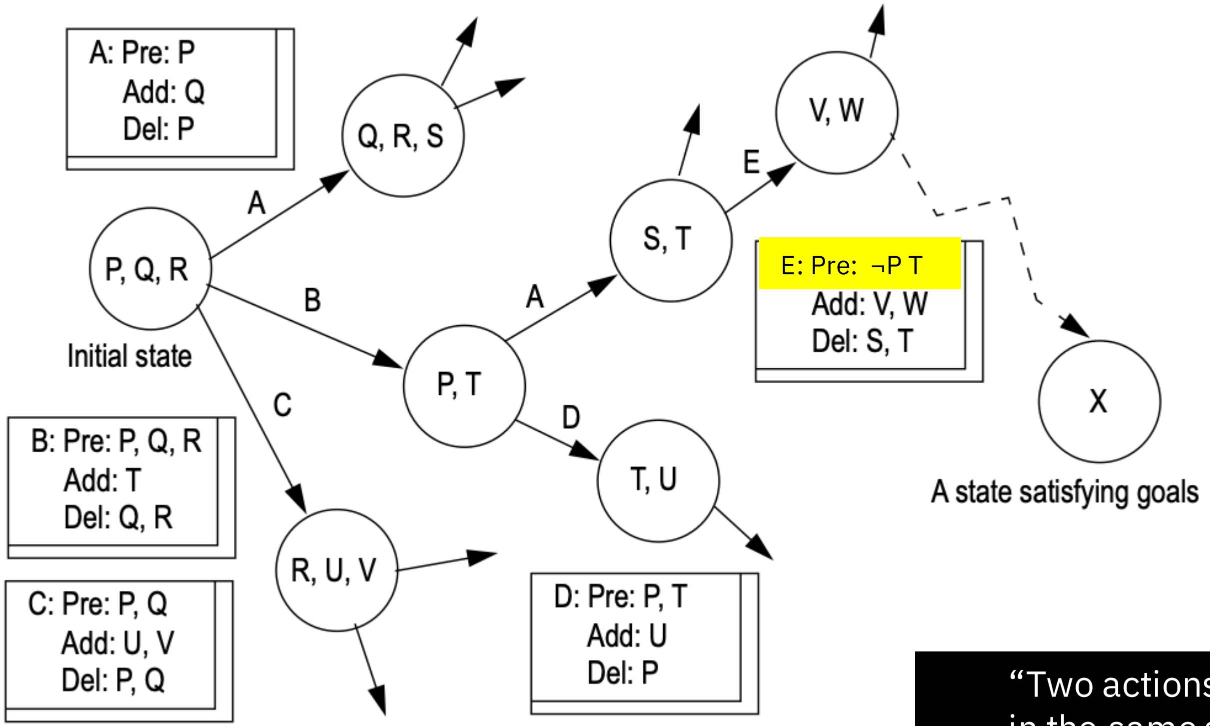
# MDP

$\langle S, A, T, R, \gamma \rangle$

## Actions

[east, west, north, south]






Long and Fox (2002)

“Two actions that are applicable in the same state cannot have the same label”

```
(:action pickup
:parameters   (?k - key ?r - room)
:precondition (and  (at ?k ?r)
      (at-agent ?r)
      (empty-hand))
:effect (and  (not (at ?k ?r))
      (not (empty-hand))
      (carry ?k))
)
```

# of grounding = #of keys * # of rooms

# Are all the parameters of LAM relevant?*

*Do they define different grounded actions that can be applied in a single state?

# Relevant parameters

Know,

(at key1 room1) $\oplus$ (at key1 room2)

So,

(pickup key1 room1) $\oplus$ (pickup key1 room2)

$\oplus$: Mutually exclusive

(**:action** pickup
**:parameters** (?k - key ?r - room)
**:precondition** (**and** (at ?k ?r)
(at-agent ?r)
(empty-hand))
**:effect** (**and** (**not** (at ?k ?r))
(**not** (empty-hand))
(carry ?k))
)

# Applicable Action Mutex Group (AAMG)

(pickup key1 room1),
(pickup key1 room2),
(pickup key1 room3),
.
.
.

⊕

(pickup key2 room1),
(pickup key2 room2),
(pickup key2 room3),
.
.
.

⊕

(pickup key1 ␣ )

(pickup key2 ␣ )

(pickup ?k - key ?r - room )

(pickup ?k – key ␣)

Parameter Seed Set of pickup
{?k – key }

# Action Space Reduction

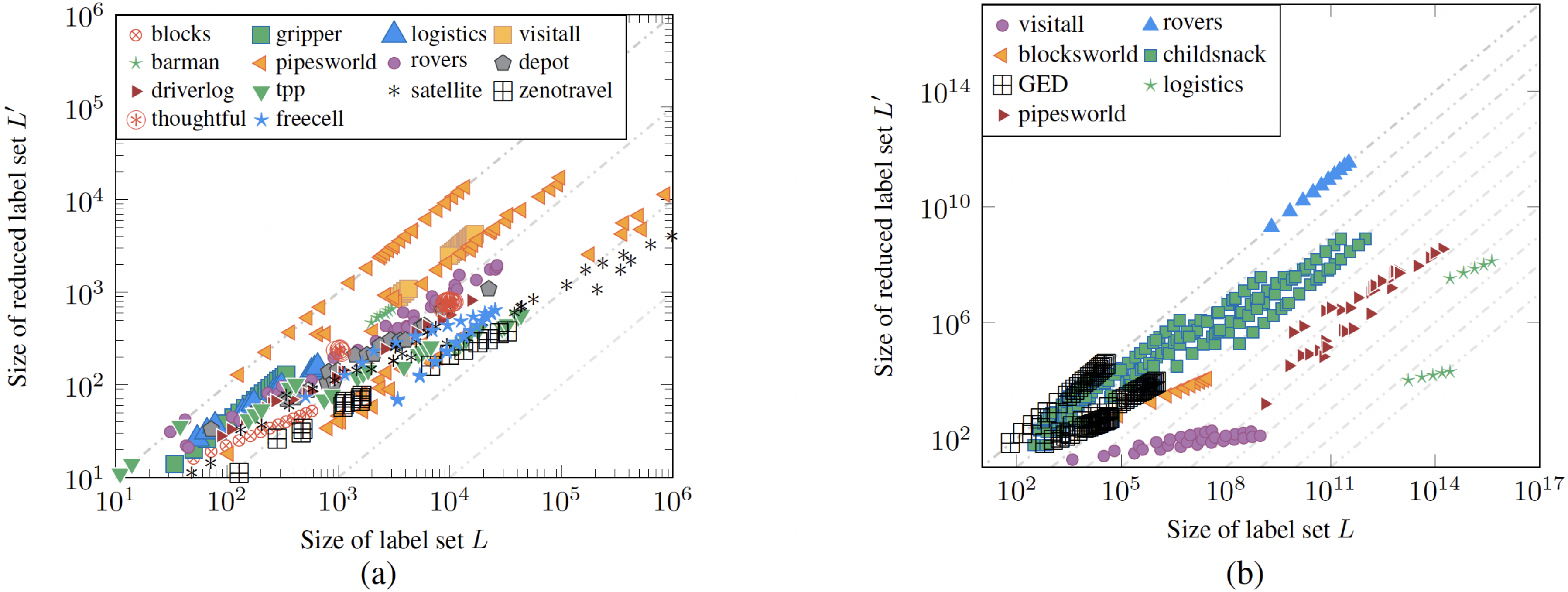



Figure 2: Comparison of label set sizes on (a) 14 IPC STRIPS domains and (b) 7 HTG domains.

# Impact on learning RL policies

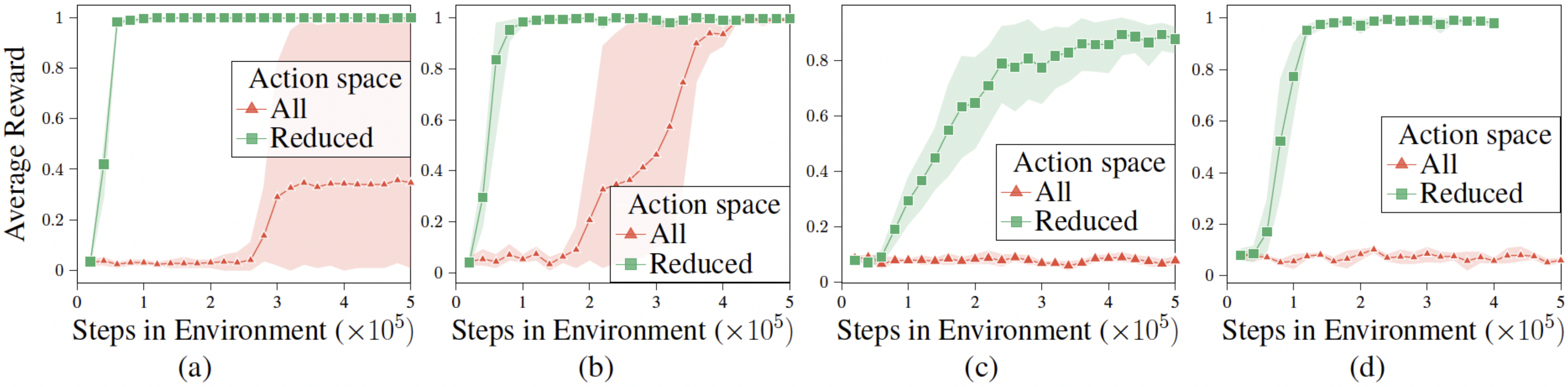
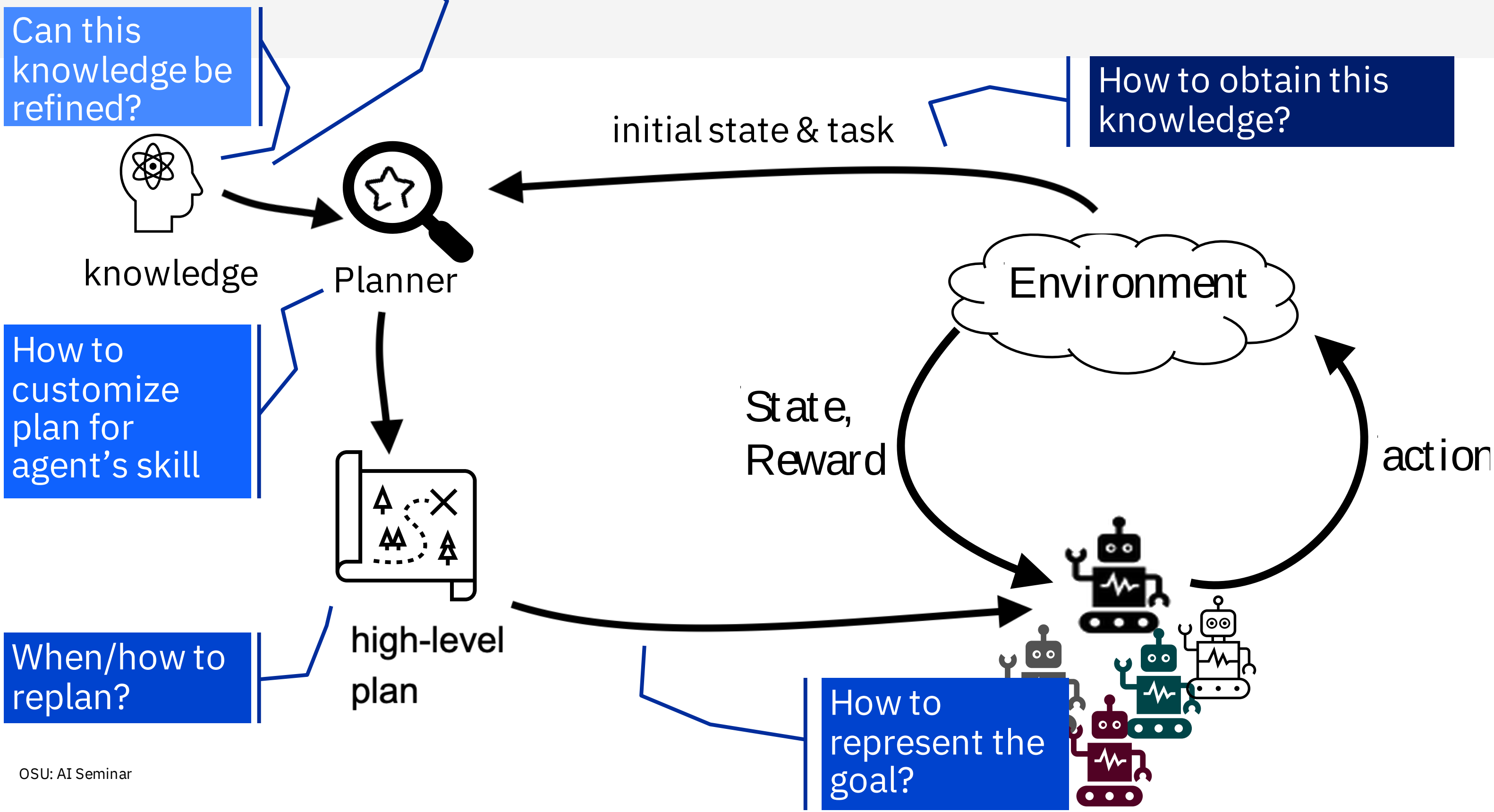



Figure 3: Learning curve in the (a) ferry, (b) gripper, (c) blocks, and (d) logistics; with and without action label reduction.

# Integrating Planning and RL



# Reference


Harsha Kokel, Arjun Manoharan, Sriraam Natarajan, Balaraman Ravindran, Prasad Tadepalli, *RePReL: Integrating Relational Planning and Reinforcement Learning for Effective Abstraction*, In **ICAPS 2021a.**

Harsha Kokel, Mayukh Das, Rakibul Islam, Julia Bonn, Jon Cai, Soham Dan, Anjali Narayan-Chen, Prashant Jayannavar, Janardhan Rao Doppa, Julia Hockenmaier, Sriraam Natarajan, Martha Palmer, Dan Roth, *Human-guided Collaborative Problem Solving: A Natural Language based Framework*, In **ICAPS (demo track) 2021b**.

Harsha Kokel, Sriraam Natarajan, Balaraman Ravindran, Prasad Tadepalli, RePReL: *A Unified Framework for Integrating Relational Planning and Reinforcement Learning for Effective Abstraction in Discrete and Continuous Domains*, In **NCAA 2022a**.

Harsha Kokel, Nikhilesh Prabhakar, Sriraam Natarajan, Balaraman Ravindran, Prasad Tadepalli, *Hybrid Deep RePReL: Integrating Relational Planning and Reinforcement Learning for Information Fusion*, In **FUSION 2022b**.

Harsha Kokel, Mayukh Das, Rakibul Islam, Julia Bonn, Jon Cai, Soham Dan, Anjali Narayan-Chen, Prashant Jayannavar, Janardhan Rao Doppa, Julia Hockenmaier, Sriraam Natarajan, Martha Palmer, Dan Roth, *Lara -- Human-guided collaborative problem solver: Effective integration of learning, reasoning and communication*, In **ACS 2022c**.

Harsha Kokel, Junkyu Lee, Michael Katz, Kavitha Srinivas, Shirin Sohrabi, Action Space Reduction for Planning Domains, IJCAI 2023

Sriraam Natarajan

Nikhilesh Prabhakar

UT DALLAS
The University of Texas at Dallas

Ravindran Balaraman

Arjun Manoharan

IIT MADRAS
Indian Institute of Technology Madras

Prasad Tadepalli

Oregon State University

Eric Blasch

AFRL
THE AIR FORCE RESEARCH LABORATORY

Junkyu Lee

Kavitha Srinivas

Michael Katz

Shirin Sohrabi

IBM Research

# Questions?

T

# Backup Slides

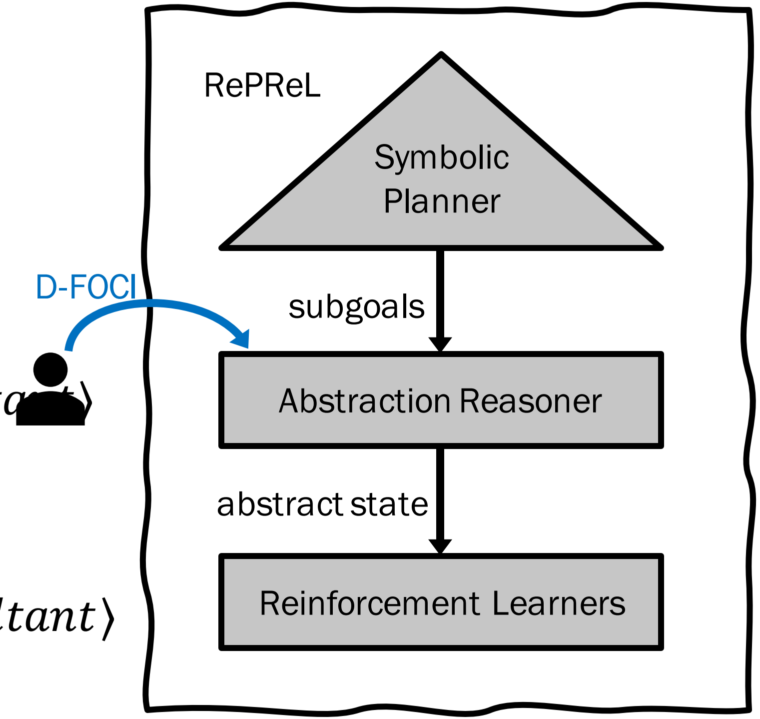
# D-FOCI

First Order Conditional Influence (FOCI) statements

$$\texttt{if}\ \langle condition \rangle\ \texttt{then}\ \langle influents \rangle\ \ \texttt{QINF}\ \langle resultant \rangle$$

Dynamic FOCI statements

$$[\langle subgoal \rangle]\text{:}\ \langle influents \rangle \xrightarrow{[+1]} \langle resultant \rangle$$




Natarajan, Tadepalli, Dietterich, and Fern 2008

# Abstraction

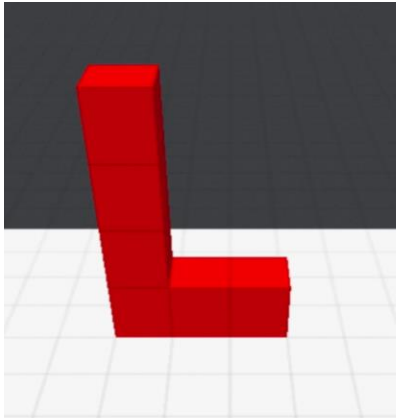
**Definition 4** (Li, Walsh, and Littman (2006)). *A model-agnostic abstraction* $\phi(s)$ *is such that for any action* $a$ *and abstract state* $\bar{s}$, $\phi(s_1) = \phi(s_2)$ *if and only if*

$$\sum_{\{s_1' | \phi(s_1') = \bar{s}\}} R_o(s_1, a, s_1') = \sum_{\{s_2' | \phi(s_2') = \bar{s}\}} R_o(s_2, a, s_2')$$

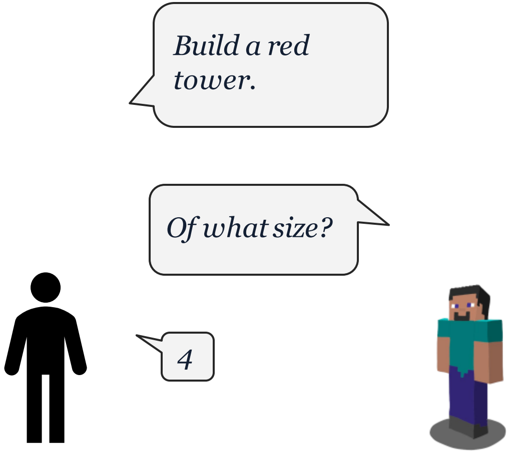
$$\sum_{\{s_1' | \phi(s_1') = \bar{s}\}} P_o(s_1, a, s_1') = \sum_{\{s_2' | \phi(s_2') = \bar{s}\}} P_o(s_2, a, s_2')$$

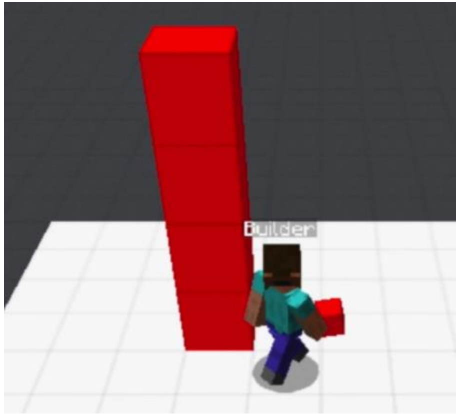
Dietterich NeurIPS 2000; Ravindran and Barto IJCAI 2003;
Givan, Dean, and Greig AI 2003; Li, Walsh, and Littman ISAIM 2006

# Collaborative Problem Solving



Target structure




Minecraft

Kokel et al. ACS 2022c

# Capabilities

- Bidirectionally contentful
  - ask for missing dimensions
  - ask for reference points
- Context-aware
  - Understand the next instruction in context of current structure
- Composable Vocabulary
  - can plan for arbitrary combinations of primitive shape
  - learn a new shape from single example
  - use learnt shape in various combinations

- Habitability
  - actionable errors/questions
- Robustness
  - Undo which doesn't lose context

# RePReL

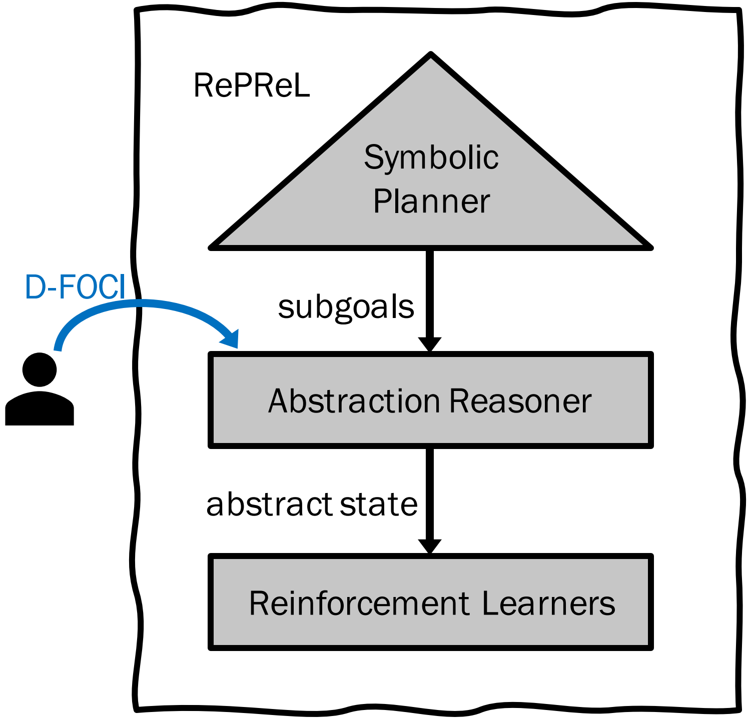
Integrating Relational Planning and Reinforcement Learning

Plan the sequence of high level subgoals and learn to execute each subgoal at lower level

Advantage:

- Compositionality
- Task specific state representations

Dynamic First Order Conditional Influence (D-FOCI) statements to obtain task-specific abstract representations




Kokel et al. ICAPS 2021

# RePReL

**Definition 3.** *The subgoal RMDP $M_o$ for each operator $o$ is defined by the tuple $\langle S, A, P_o, R_o, \gamma \rangle$ consisting of states $S$, actions $A$, transition function $P_o$, reward function $R_o$, and discount factor $\gamma$. State and Actions remain same as the original RMDP. The reward function $R_o$ and transition probability distribution function $P_o$ are defined as follows:*

$$R_o(s,a,s') = \begin{cases} t_R + R(s,a,s') & \textit{if } s' \in \beta(o) \textit{ and } s \notin \beta(o) \\ 0 & \textit{if } s' \in \beta(o) \textit{ and } s \in \beta(o) \\ R(s,a,s') & \textit{otherwise} \end{cases}$$

$$P_o(s,a,s') = \begin{cases} 0 & \textit{if } s \in \beta(o) \textit{ and } s' \notin \beta(o) \\ 1 & \textit{if } s \in \beta(o) \textit{ and } s' \in \beta(o) \\ P(s,a,s') & \textit{otherwise} \end{cases}$$

*with $R(s,a,s')$ indicating the reward function from the original GRMDP definition. $t_R$ is a fixed terminal reward.*

# Abstraction

**Given:**

a. D-FOCI statements from Equation 3

b. state $s = \{$ **at**(p1,r), **taxi_at**(l3), **dest**(p1,d1), $\neg$**at_dest**(p1) $\neg$**in_taxi**(p1), **at**(p2,b), $\neg$**at_dest**(p2), $\neg$**in_taxi**(p2)$\}$

c. grounded option$o\theta$: **pick**($P$) $\{P/\text{p1}\}$

**Output:** A set of relevant state literals: $\hat{s}$

**Depth 1 unrolling**:

1. Find a substitution that grounds relevant D-FOCI statements that have reward on RHS

   **pick**(p1): **in_taxi**(p1) $\longrightarrow$ *Reward*

   $\theta = \{\text{P/p1}\}$

2. Collect LHS in relevant literals set $\hat{s}$

   $\hat{s} \leftarrow \{$**in_taxi**(p1)$\}$

**Depth 2 unrolling**:

1. Find a substitution that grounds relevant D-FOCI statements that have a relevant literal on RHS

   **pick**(P): $\{$ **action**, **taxi_at**(l3), **at**(p1, r), **in_taxi**(p1) $\}$ $\longrightarrow$ **in_taxi**(p1)

   $\theta = \{\text{P/p1, X/l3, Y/r}\}$

2. Collect LHS in set $\hat{s}$

   $\hat{s} \leftarrow \{$**in_taxi**(p1), **action**, **taxi_at**(l3), **at**(p1, r)$\}$

**Depth 3 unrolling:**

1. Ground applicable D-FOCI statements that have a relevant literal ($\hat{s}$) on RHS

   $\{$**action**, **taxi_at**(l3) $\}$ $\xrightarrow{+1}$ **taxi_at**(l3)

   **pick**(p1): $\{$ **action**, **taxi_at**(l3), **at**(p1, r), **in_taxi**(p1) $\}$ $\longrightarrow$ **in_taxi**(p1)

   $\theta = \{\text{P/p1, X/l3, Y/r}\}$

2. Collect LHS in set $\hat{s}$

   $\hat{s} \leftarrow \{$**in_taxi**(p1), **action**, **taxi_at**(l3), **at**(p1, r)$\}$

recursive grounding and unrolling process

# Lifted Successor Generation

Treats planning state as database and task of generating applicable action groundings as a database query.

```
(at obj1 l1)
(at obj2 l1)
(at obj3 l3)
(at obj4 l2)
(path l1 l2)
(path l1 l3)
(path l2 l3)
(path l3 l4))
```

| *at* | |
|---|---|
| obj1 | l1 |
| obj2 | l1 |
| obj3 | l3 |
| obj4 | l2 |

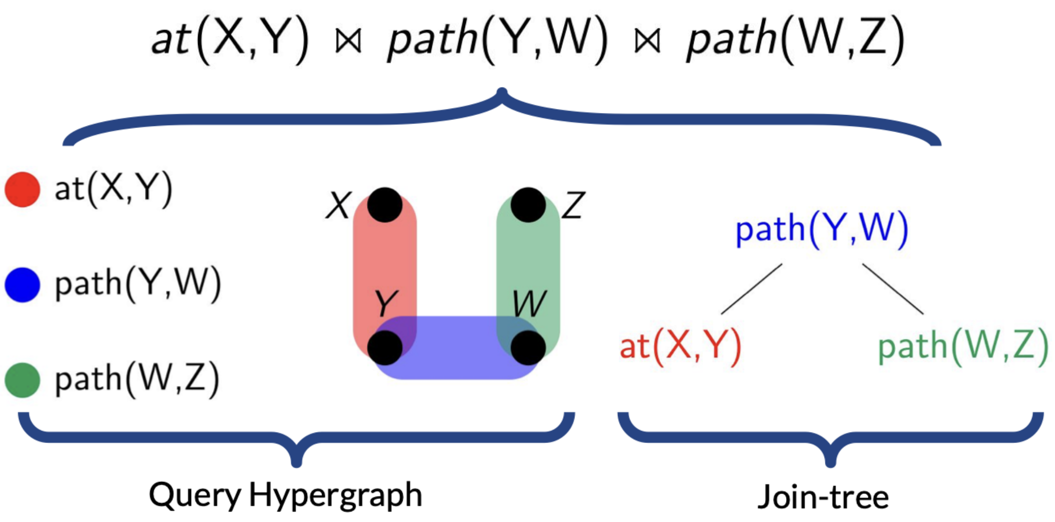
| *path* | |
|---|---|
| l1 | l2 |
| l1 | l3 |
| l2 | l3 |
| l3 | l4 |

```
(:precondition
(and (at ?X ?Y)
     (path ?Y ?W)
     (path ?W ?Z)))
```

$$\underline{\mathrm{at}(X, Y)} \bowtie \underline{\mathrm{path}(Y, W)} \bowtie \underline{\mathrm{path}(W, Z)}$$

Augusto Corrêa et al. 2020

# Query Evaluation




**If query hypergraph has a join-tree, it is acyclic**
**Acyclic query evaluation is output-polynomial**

If hypergraph is cyclic, the query evaluation is exponential in the size of input and output.

With parameter seed set we can improve cyclic query evaluation time.

# Query Evaluation w/ Seed Set

$$Q(X,Y,W,Z) = \mathrm{at}(X,Y) \bowtie \mathrm{path}(Y,W) \bowtie \mathrm{path}(W,Z) \bowtie \mathrm{path}(Y,Z) \bowtie \mathrm{path}(Z,X)$$

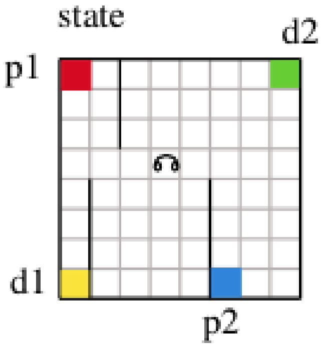
Treat the non-seed parameters as *non-distinguishable* variable

$$Q(X,W,Z) = \mathrm{at}(X,Y) \bowtie \mathrm{path}(Y,W)$$
$$\bowtie \mathrm{path}(W,Z) \bowtie \mathrm{path}(Y,Z) \bowtie \mathrm{path}(Z,X)$$

Modify the join order

$$Q(X,Y,W,Z) = \mathrm{path}(Y,Z) \bowtie \mathrm{at}(X,Y)$$
$$\bowtie \mathrm{path}(Y,W) \bowtie \mathrm{path}(W,Z) \bowtie \mathrm{path}(Z,X)$$

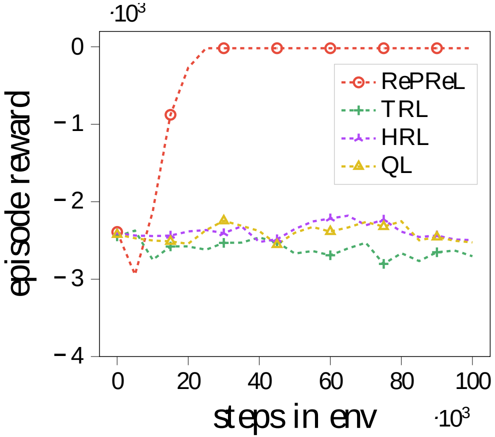
# Experiments



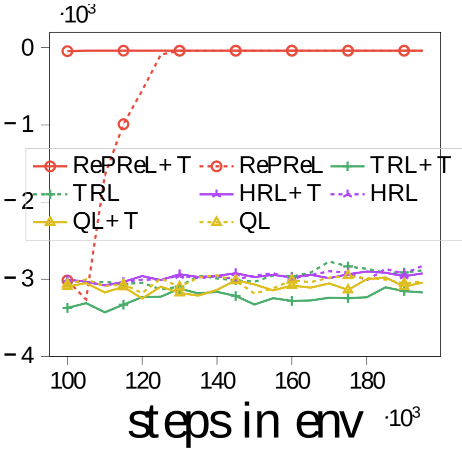

Sample efficiency

Transfer across task

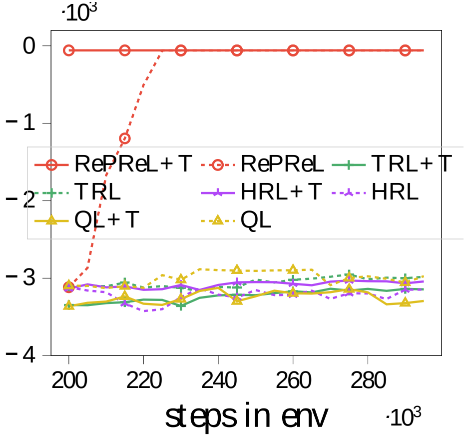
Generalization across objects




transport p1



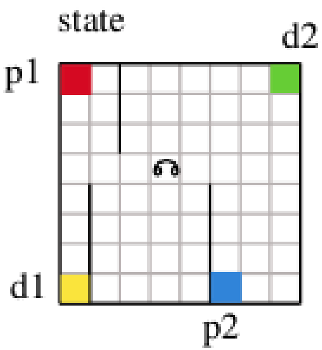

transport p1 and p2

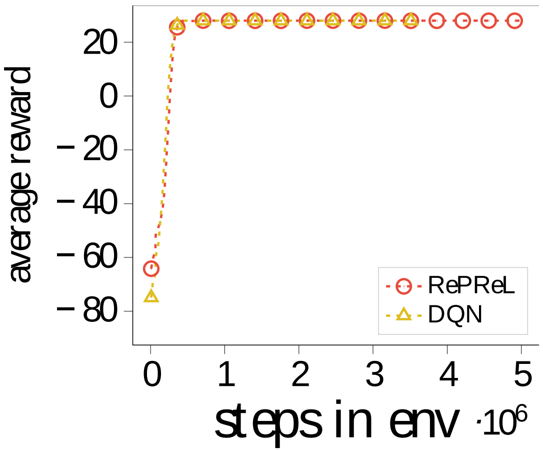



transport p1, p2 and p3

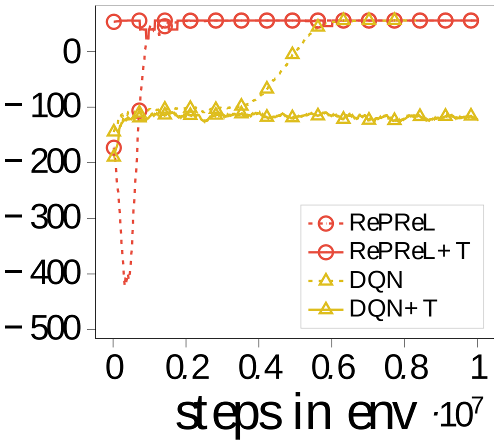
Kokel et al. ICAPS 2021

# Deep Relational RL

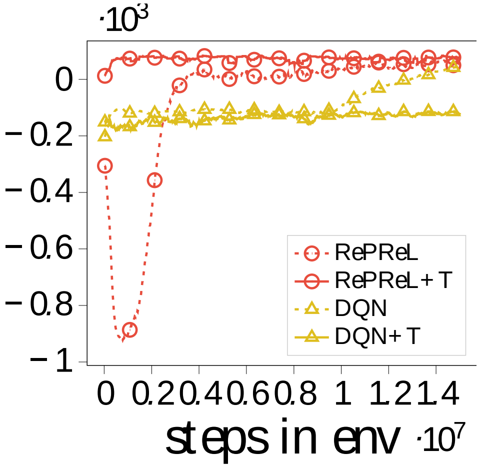



transport p1




transport p1 and p2



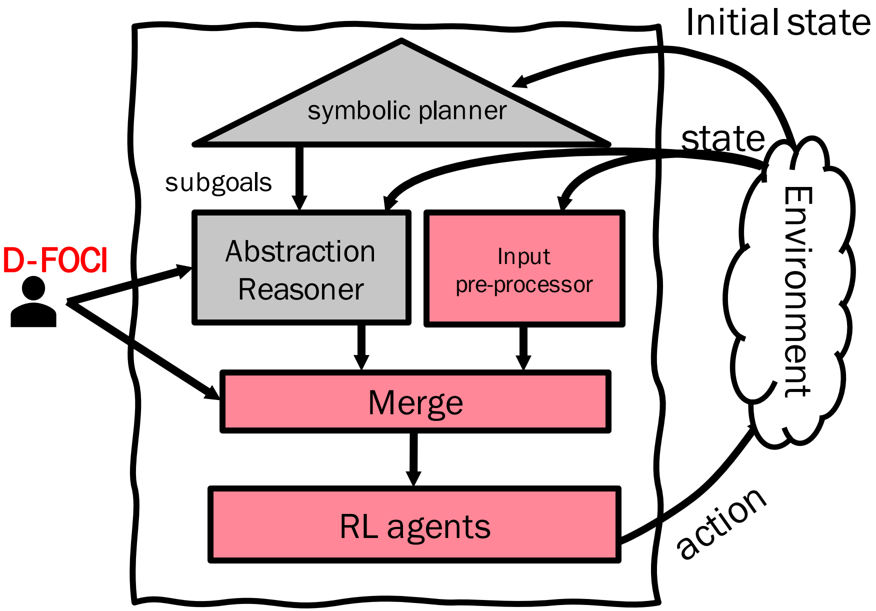

transport p1, p2 and p3

Kokel et al. NCAA 2022a

# Extension to hybrid domain

Allow hybrid of structured and unstructured data

Neural predicates in D-FOCI



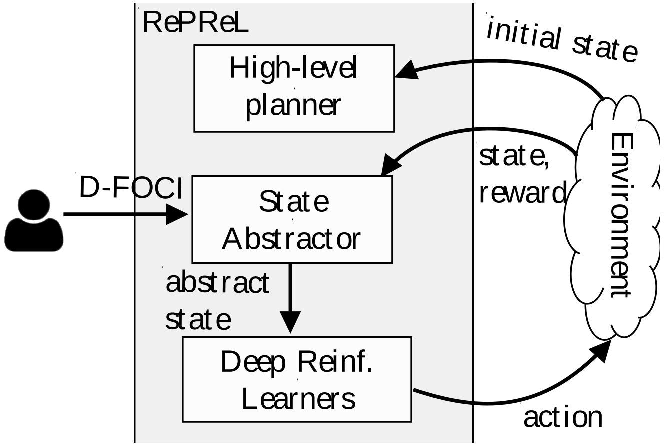

Kokel et al. FUSION 2022b

# Extension to deep, relational RL

Allowed continuous state and action space

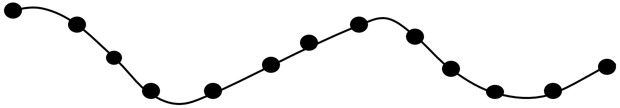
Batch learning allowed any off-policy RL agent (inc. deep relational RL)

Removed fixed depth unrolling limitation

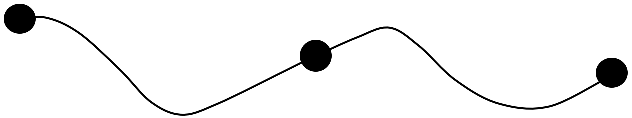



Kokel et al. NCAA 2022a

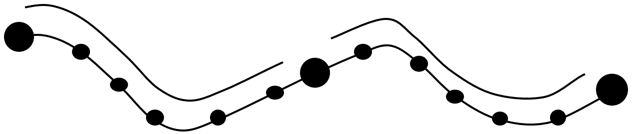
Markov Decision Process
MDP



Semi-MDP

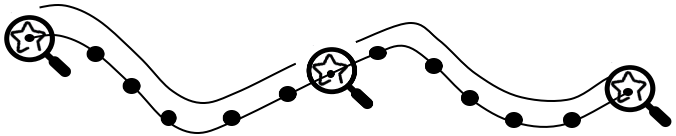


Hierarchical RL or
Options



$$O = \langle I, \beta, \pi \rangle$$

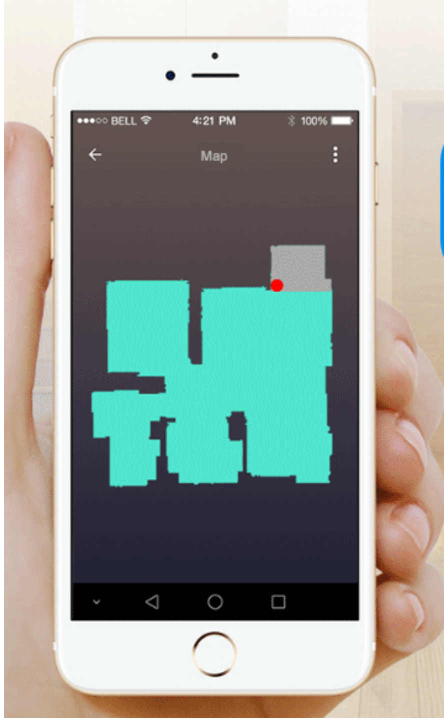
Planning
with RL



# Reinforcement Learning

Skinner, B. F. (1961). Teaching machines.

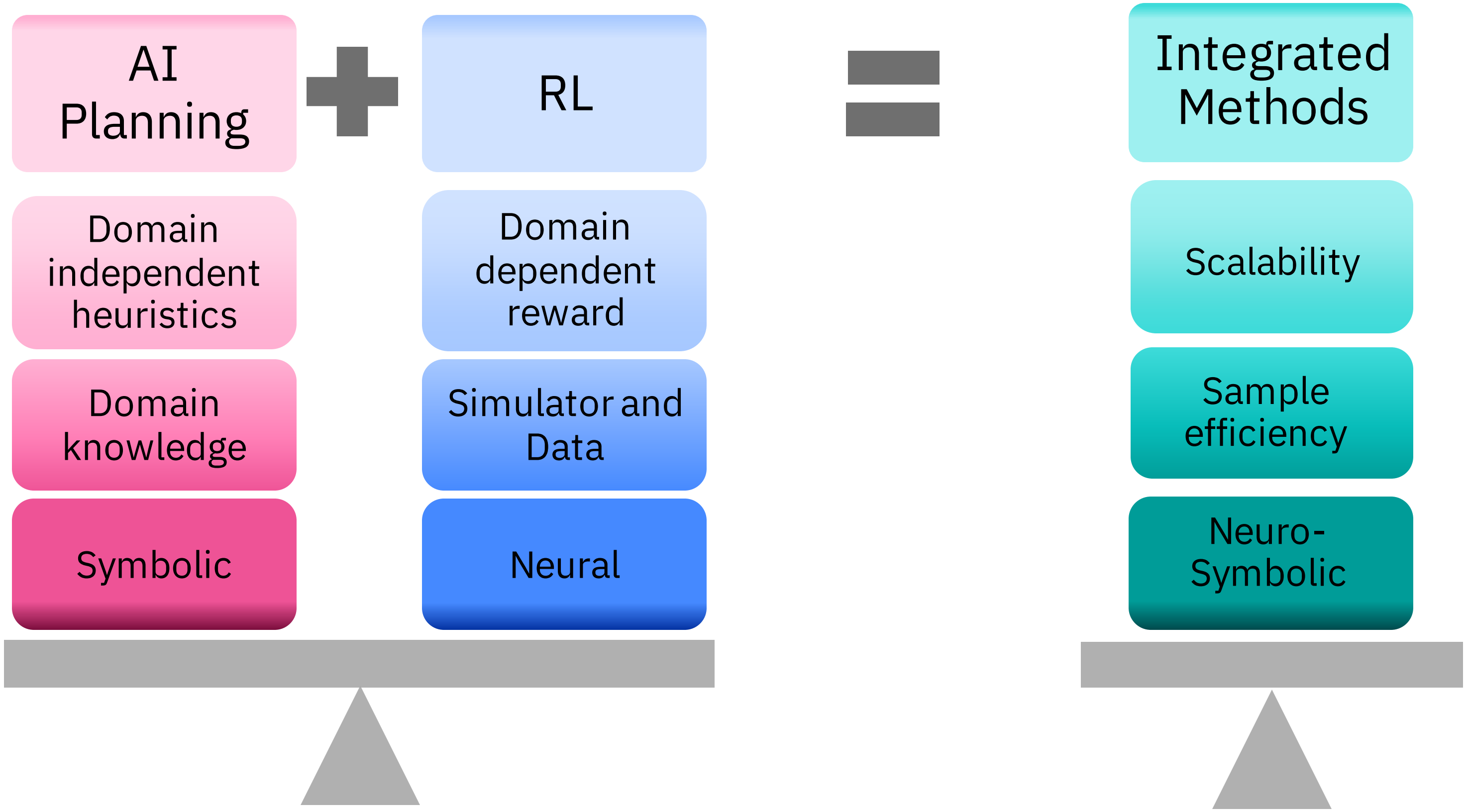
# Compositional and Relational Domains



# Compositional and Relational Domains

# Compositional and Relational Domains

# Sequential decision making



# LMG to identify Seed Set

$$\ell = \langle \{?\mathrm{k} - \mathrm{object}\}, \{?\mathrm{r} - \mathrm{location}\}, \{(\mathrm{at}\ ?\mathrm{k} - \mathrm{object}\ \ ?\mathrm{r} - \mathrm{location})\} \rangle$$

Conditions

1. atom of LMG is part of precondition
2. variable types in LMG is super-type of variable type of action parameter

Then removing the *counted variable* of LMG from action parameter defines an AOMG.

So, counted variable is <u>not</u> a seed set.

Leverage multiple LMGs in sequence to further reduce the parameter set.

(**:action** pickup

**:parameters** (?k - key ?r - room)

**:precondition** (**and** (at ?k ?r)

(at-agent ?r)

(empty-hand))

**:effect** (**and** (**not** (at ?k ?r))

(**not** (empty-hand))

(carry ?k))

)

# Lifted Mutex Group (LMG)

- ❑ Mutex Group

A set of facts, of which only one fact is true in any reachable state

Example:

{ (at key1 room1), (at key1 room2) ,
(at key1 room3), (at key1 room4) }

- ❑ Lifted Mutex Group[1]

An invariant candidate, whose *ground atom sets* are mutex groups

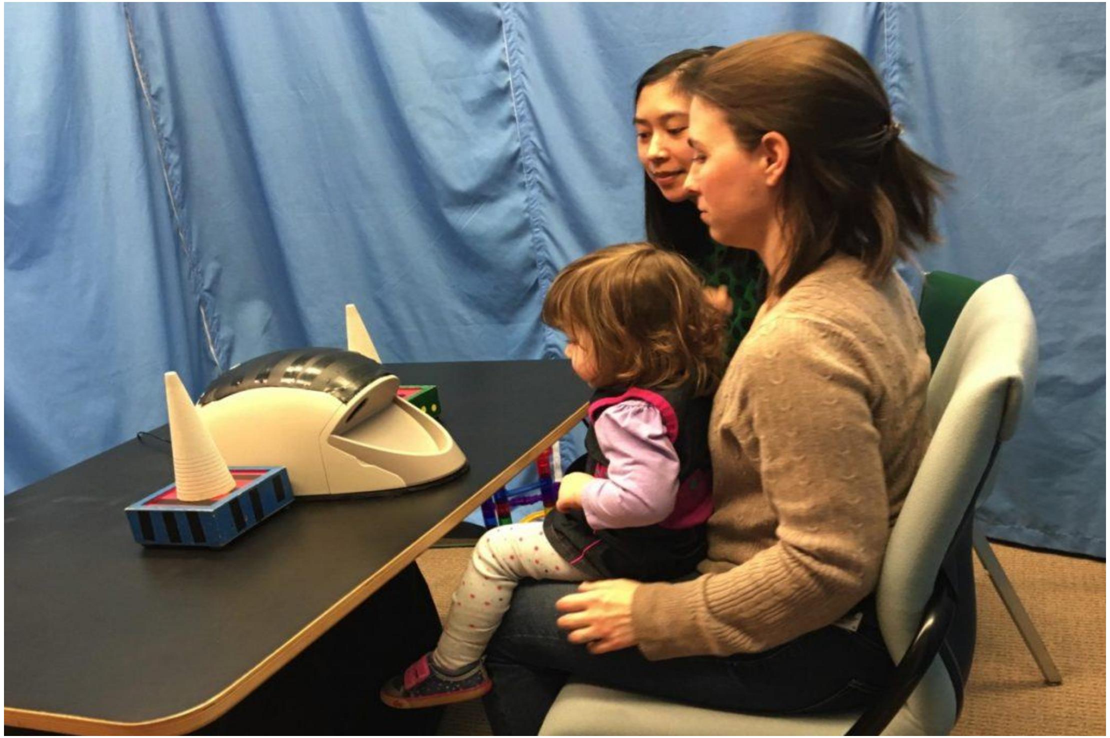
Example: fixed variables counted variables

$\ell = \langle \{?\mathrm{k} - \mathrm{key}\}, \{?\mathrm{r} - \mathrm{room}\}, \{(\mathrm{at}\ ?\mathrm{k} - \mathrm{key}\ \ ?\mathrm{r} - \mathrm{room})\} \rangle$

$\ell$ (?k/key1) = { (at key1 room1), (at key1 room2) ,
(at key1 room3), (at key1 room4) }

$\ell$ (?k/key2) = { (at key2 room1), (at key2 room2) ,
(at key2 room3), (at key2 room4) }

[1]Dan Fiser 2020



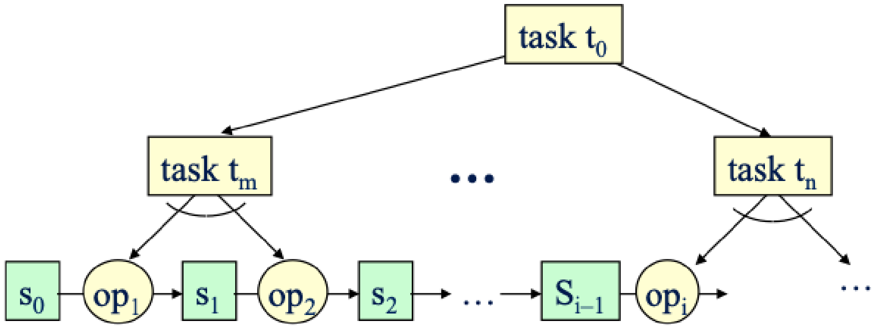
Meltzoff, Waismeyer, & Gopnik (2012)

# Hierarchical Planning

Planning for *Tasks* instead of goals

*Methods* to decompose tasks

*Operators* for executing action



# Hierarchical Planning

Domain has predicates (Q), operators (O) and methods (M)

Methods have preconditions and subtasks

Operators have preconditions and effects

$$\mathcal{D} = \langle Q, O, M \rangle$$

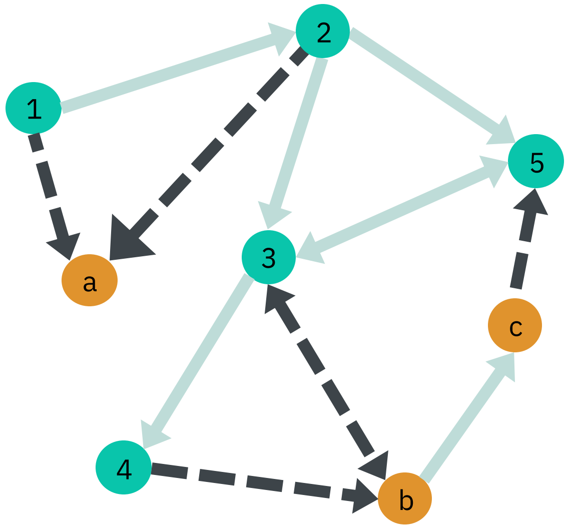
Planning problem

$$m = \langle task(m), pre(m), subtasks(m) \rangle$$

$$o = \langle task(o), pre(o), eff(o) \rangle$$

$$\mathcal{P} = \langle \mathcal{D}, \text{initial state } s_0, \text{task list } t_o \rangle$$

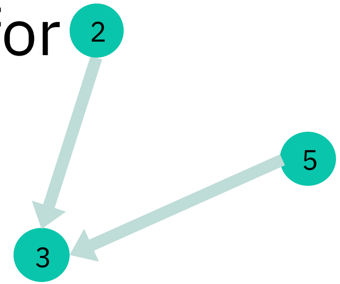
# Example



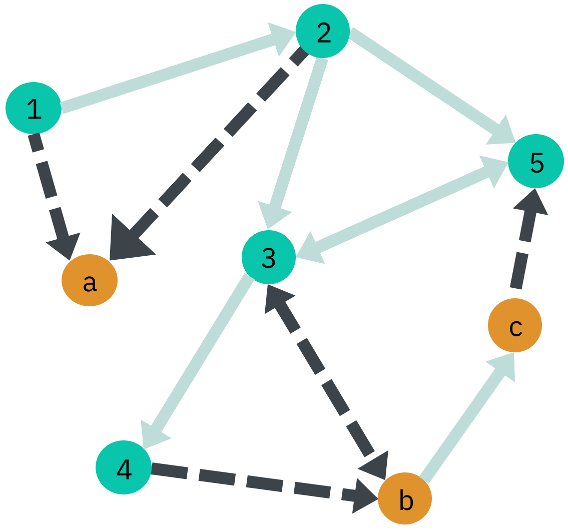

task1(X): solid(Y, G) $\xrightarrow{+1}$ color(G, green)

task1(X): color(X, green) $\xrightarrow{+1}$ Reward

Abstract state for task1(3):



# Example




task1(X): dashed(G, Z) $\xrightarrow{+1}$ color(G, green)

task1(X): solid(Y, G) $\xrightarrow{+1}$ color(G, green)

task1(X): color(X, green) $\xrightarrow{+1}$ Reward

Abstract state for task1(3):