LTL2Action

Generalizing LTL Instructions for Multi-Task RL




Pashootan Vaezipoor *Andrew Li* *Rodrigo Toro Icarte* *Sheila McIlraith*

UNIVERSITY OF TORONTO
VECTOR INSTITUTE | INSTITUT VECTEUR
CIFAR
SCHWARTZ REISMAN INSTITUTE FOR TECHNOLOGY AND SOCIETY

Motivation

- A long-standing aspiration of AI is to build agents that can understand and follow human instructions to solve problems. **[McCarthy et al., 1960]**

- Task specification:
  - **Reward function:** Hard to specify for each task
  - **Natural language:** Hard to map to a reward for every possible environment

Motivation

- A long-standing aspiration of AI is to build agents that can understand and follow human instructions to solve problems. **[McCarthy et al., 1960]**
- Task specification:
  - **Reward function:** Hard to specify for each task
  - **Natural language:** Hard to map to a reward for every possible environment
- We express instructions in formal language of *linear temporal logic (LTL)*

Motivation

- A long-standing aspiration of AI is to build agents that can understand and follow human instructions to solve problems. **[McCarthy et al., 1960]**
- Task specification:
  - **Reward function:** Hard to specify for each task
  - **Natural language:** Hard to map to a reward for every possible environment
- We express instructions in formal language of *linear temporal logic (LTL)*

Motivation

- A long-standing aspiration of AI is to build agents that can understand and follow human instructions to solve problems. **[McCarthy et al., 1960]**
- Task specification:
  - **Reward function:** Hard to specify for each task
  - **Natural language:** Hard to map to a reward for every possible environment
- We express instructions in formal language of *linear temporal logic (LTL)*

$$\pi(a|s,\varphi)$$

Instruction
(task)

Motivation

- A long-standing aspiration of AI is to build agents that can understand and follow human instructions to solve problems. **[McCarthy et al., 1960]**

- Task specification:
  - **Reward function:** Hard to specify for each task
  - **Natural language:** Hard to map to a reward for every possible environment

- We express instructions in formal language of *linear temporal logic (LTL)*
  - **Expressiveness:** Temporal modalities

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \psi \mid \overset{\texttt{Next}}{\bigcirc}\varphi \mid \varphi \underset{\texttt{Until}}{\mathsf{U}} \psi \mid \overset{\texttt{Eventually}}{\Diamond}\varphi \mid \underset{\texttt{Always}}{\Box}\varphi$$

Motivation

- A long-standing aspiration of AI is to build agents that can understand and follow human instructions to solve problems. **[McCarthy et al., 1960]**

- Task specification:
  - **Reward function:** Hard to specify for each task
  - **Natural language:** Hard to map to a reward for every possible environment

- We express instructions in formal language of *linear temporal logic (LTL)*
  - **Expressiveness:** Temporal modalities
  - **Unambiguous semantics:** instructions are automatically mapped to reward

Motivation

- A long-standing aspiration of AI is to build agents that can understand and follow human instructions to solve problems. [McCarthy et al., 1960]

- Task specification:
  - **Reward function:** Hard to specify for each task
  - **Natural language:** Hard to map to a reward for every possible environment

- We express instructions in formal language of *linear temporal logic (LTL)*
  - **Expressiveness:** Temporal modalities
  - **Unambiguous semantics:** instructions are automatically mapped to reward
  - **Compositional syntax:** can be used to procedurally generate novel tasks

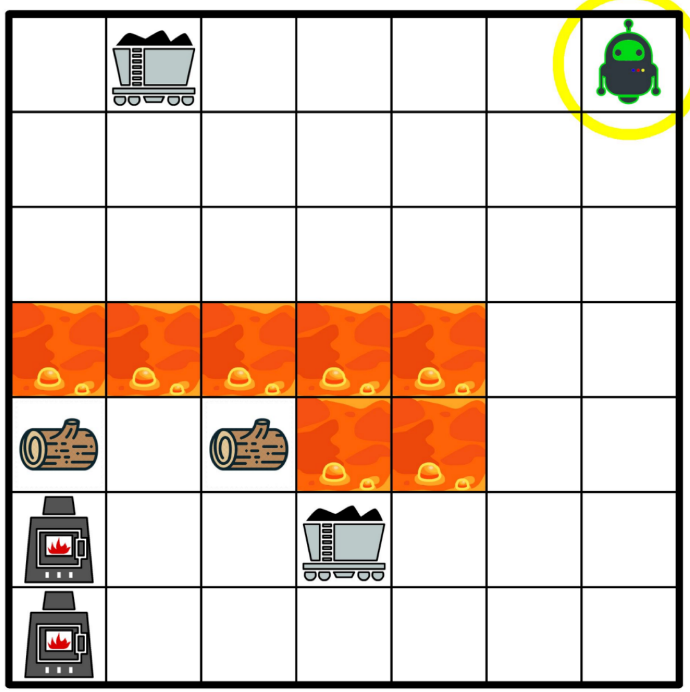
Motivation

- A long-standing aspiration of AI is to build agents that can understand and follow human instructions to solve problems. **[McCarthy et al., 1960]**
- Task specification:
  - **Reward function:** Hard to specify for each task
  - **Natural language:** Hard to map to a reward for every possible environment
- We express instructions in formal language of *linear temporal logic (LTL)*
  - **Expressiveness:** Temporal modalities
  - **Unambiguous semantics:** instructions are automatically mapped to reward
  - **Compositional syntax:** can be used to procedurally generate novel tasks
- **Theoretical** advantages:
  - Non-myopia
  - Capable of handling non-Markovian rewards

Motivation

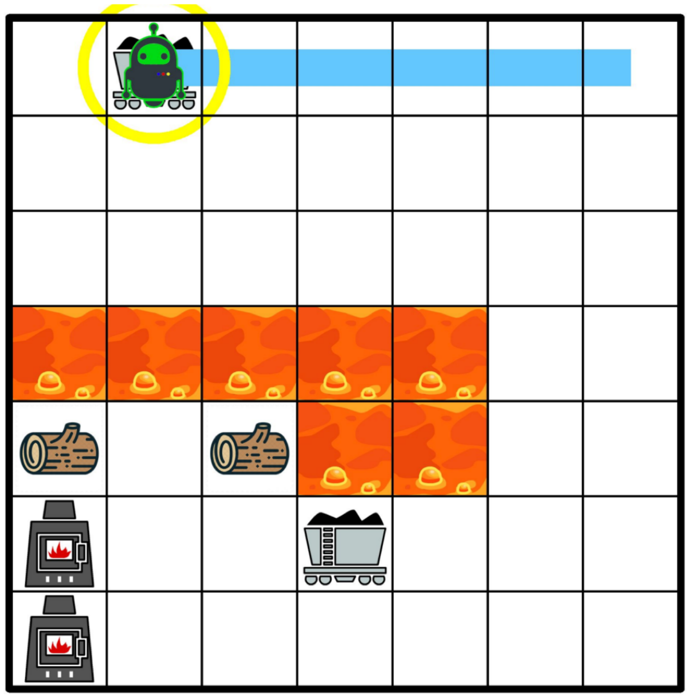
- A long-standing aspiration of AI is to build agents that can understand and follow human instructions to solve problems. [McCarthy et al., 1960]
- Task specification:
  - **Reward function:** Hard to specify for each task
  - **Natural language:** Hard to map to a reward for every possible environment
- We express instructions in formal language of *linear temporal logic (LTL)*
  - **Expressiveness:** Temporal modalities
  - **Unambiguous semantics:** instructions are automatically mapped to reward
  - **Compositional syntax:** can be used to procedurally generate novel tasks
- **Theoretical** advantages:
  - Non-myopia
  - Capable of handling non-Markovian rewards
- **Empirical** advantages:
  - Discrete and Continuous domains
  - Zero-shot generalization to unseen tasks

The Idea



Primitive events:
(Propositions)
- `pickup_coal`
- `pickup_wood`
- `use_furnace`
- `on_lava`

The Idea



Primitive events:

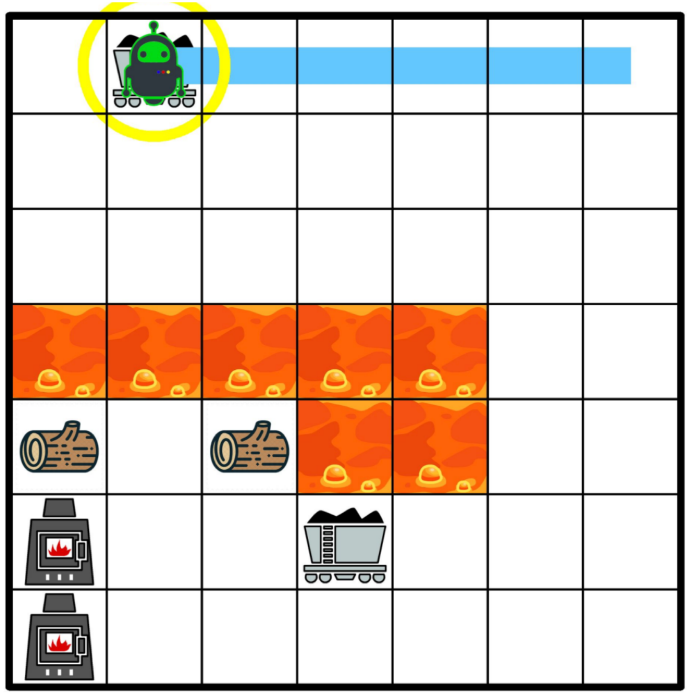
(Propositions)

- `pickup_coal`
- `pickup_wood`
- `use_furnace`
- `on_lava`

The Idea

Task: "Get coal or wood, in any order, then used the furnace."

`eventually ((pickup_coal or pickup_wood) and (eventually use_furnace))`

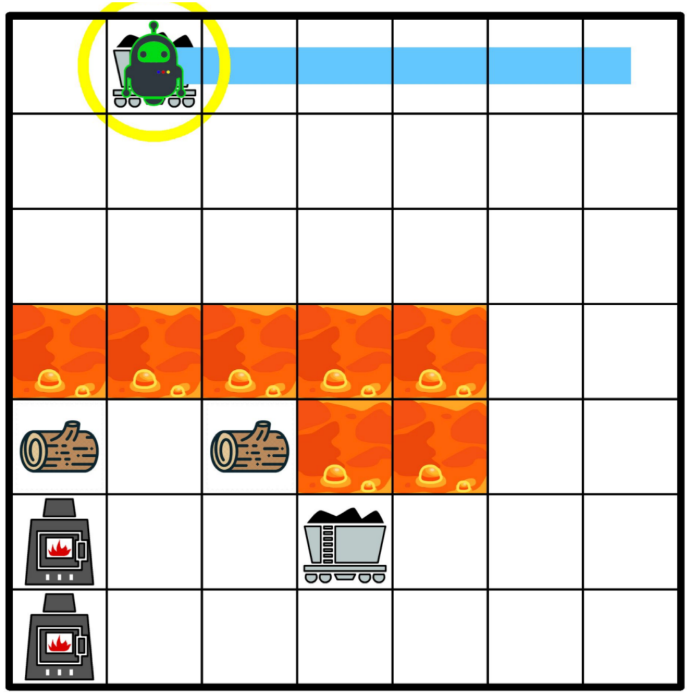


Primitive events:
(Propositions)
- `pickup_coal`
- `pickup_wood`
- `use_furnace`
- `on_lava`

The Idea

Task: "Get coal or wood, in any order, then used the furnace."

`eventually ((pickup_coal or pickup_wood) and (eventually use_furnace))`



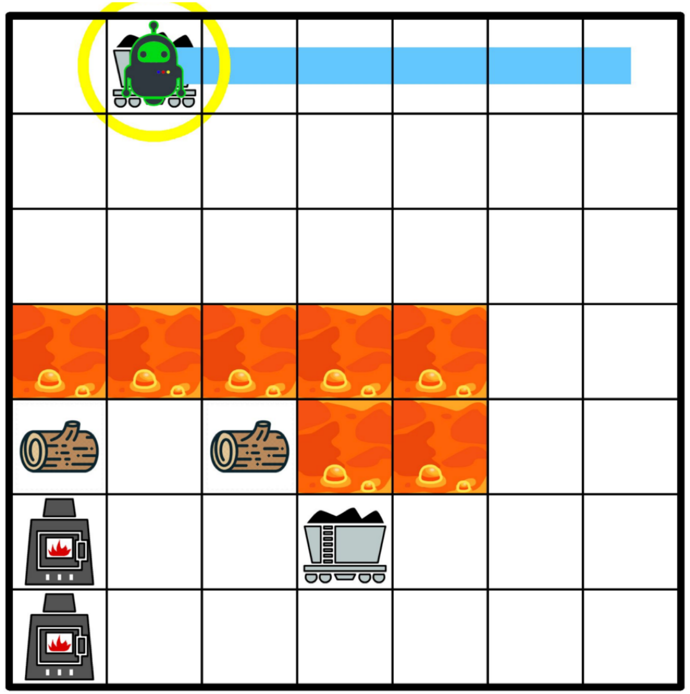
Primitive events:
(Propositions)
- `pickup_coal`
- `pickup_wood`
- `use_furnace`
- `on_lava`

$$R = \begin{cases} 1 & \text{if } \varphi \text{ is \textbf{satisfied}} \\ -1 & \text{if } \varphi \text{ is \textbf{falsified}} \\ 0 & \text{otherwise} \end{cases}$$

The Idea

Task: "Get coal or wood, in any order, then used the furnace."

`eventually ((pickup_coal or pickup_wood) and (eventually use_furnace))`



Primitive events:
(Propositions)
- `pickup_coal`
- `pickup_wood`
- `use_furnace`
- `on_lava`

$$R = \begin{cases} 1 & \text{if } \varphi \text{ is } \textbf{satisfied} \\ -1 & \text{if } \varphi \text{ is } \textbf{falsified} \\ 0 & \text{otherwise} \end{cases}$$

This reward scheme is non-Markovian!

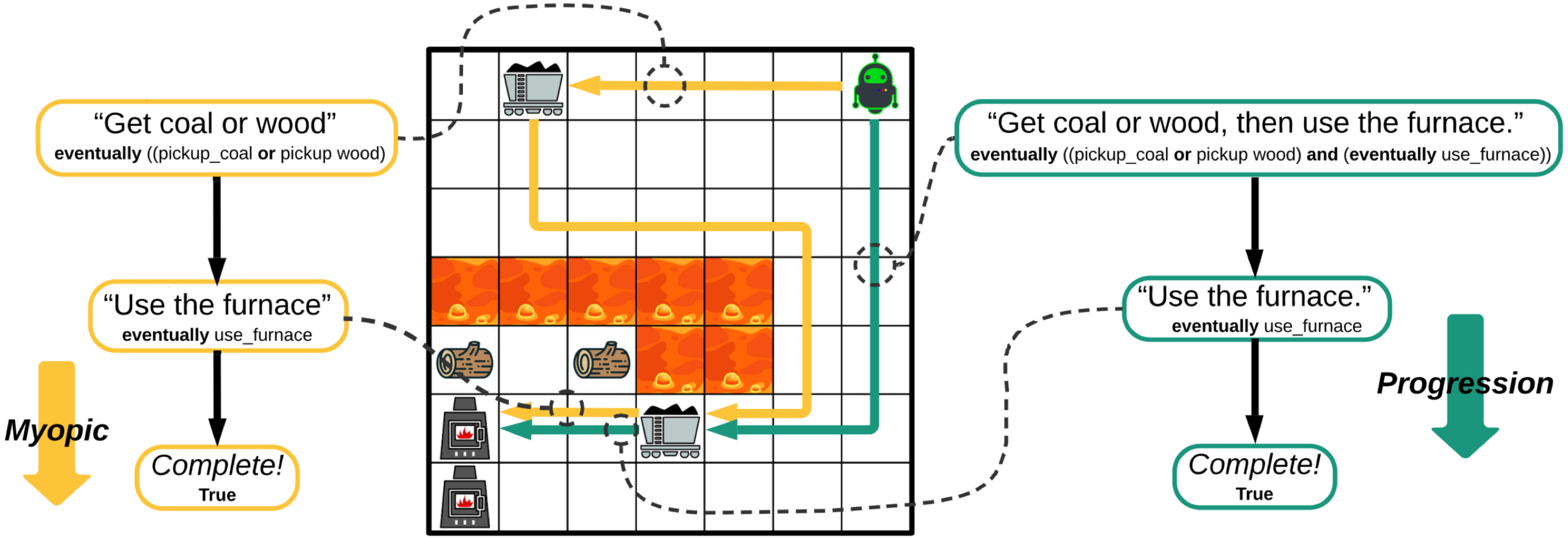
The Idea

Task decomposition

- Decompose tasks to subtasks that can be individually solved

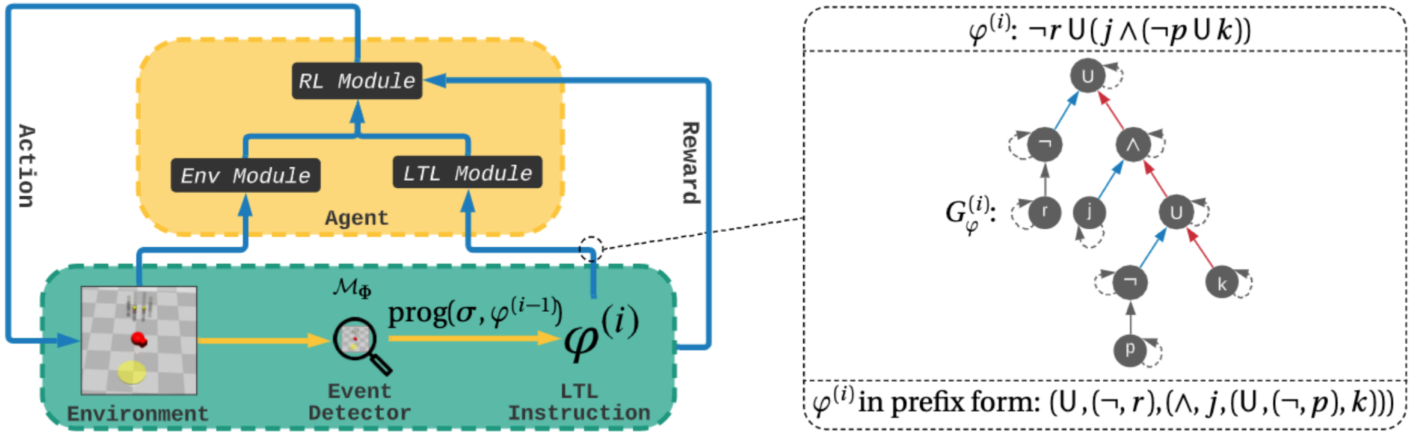
leads to suboptimal policies!

- We use LTL Progression [Bacchus Kabanza, 2000] to *automatically* simplify the instructions over time as parts of the task are solved



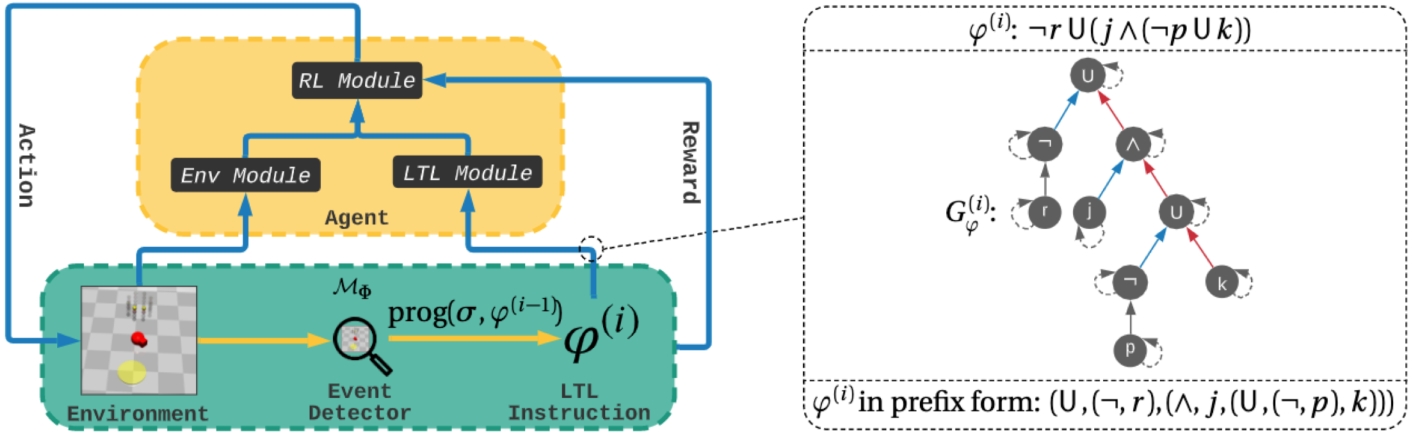
Architecture

$$\pi(a|s,\varphi)$$



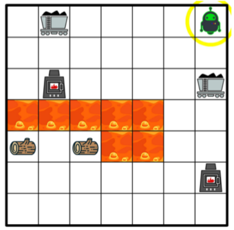
- It's easy to incorporate these concepts into a standard RL framework (e.g., PPO)

Architecture

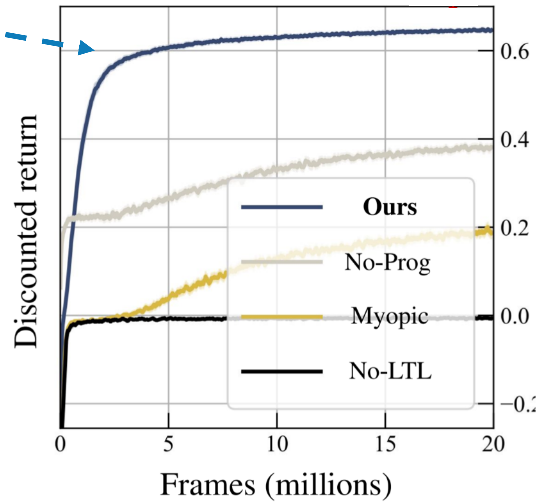


- It’s easy to incorporate these concepts into a standard RL framework (e.g., PPO)
- Key Results:
  - We outperform other approaches that do not use LTL progression or are myopic
  - Compositional architecture (GNN) encode formulae better than seq models
  - We generalize to *unseen* (and more complex) instructions than those in training

Results

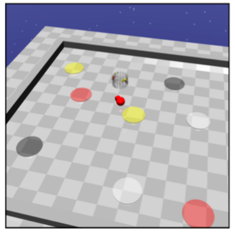


Ours

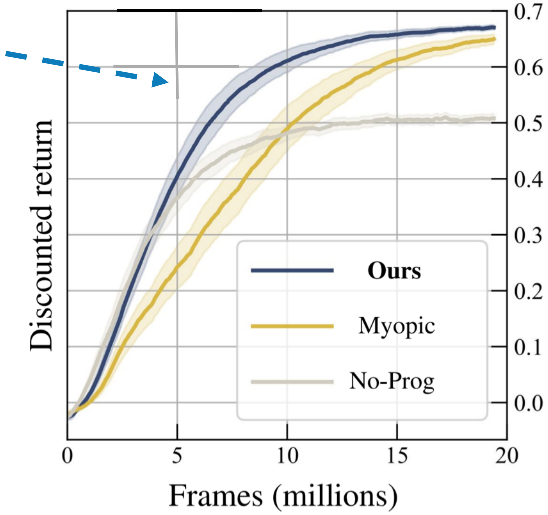


Gridworld

(Discrete)



Ours



MuJoCo

(Continuous)

Ideas for Future Work

- Remove our reliance on the *event detectors*
  - "Noisy Symbolic Abstractions for Deep RL: A case study with Reward Machines"
    [Li, Chen, PV, Klassen, Icarte, McIlraith, Deep RL Workshop 2022]
  - "Learning to Follow Instructions in Text-Based Games"
    [Tuli, Li, PV, Klassen, Sanner, McIlraith, Neurips 2022]

Ideas for Future Work

- Remove our reliance on the *event detectors*
  - "Noisy Symbolic Abstractions for Deep RL: A case study with Reward Machines"
    [Li, Chen, PV, Klassen, Icarte, McIlraith, Deep RL Workshop 2022]
  - "Learning to Follow Instructions in Text-Based Games"
    [Tuli, Li, PV, Klassen, Sanner, McIlraith, Neurips 2022]

Next Talk!

Thank you!