Beyond Fixed Tasks: Unsupervised Environment Design for Task-Level Pairs

Daniel Furelos-Blanco[1], Charles Pert[1], Frederik Kelbel[1], Alex F. Spies[1], Alessandra Russo[1], Michael Dennis[2]

[1]Imperial College London

[2]Google DeepMind

AAAI-26 / IAAI-26 / EAAI-26
JANUARY 20–27, 2026 | SINGAPORE

Acknowledgments

Charles Pert

Frederik Kelbel

Alex F. Spies

Alessandra Russo

Michael Dennis

Introduction

Training Generally Capable Agents

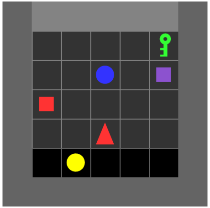
Goal

Effectively train RL agents to follow diverse instructions (**tasks**) in varied environments (**levels**)

Task

"go to a ball, then go to a red square"

Level



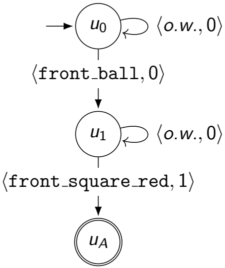
Agent: ▲

Training Generally Capable Agents

Goal

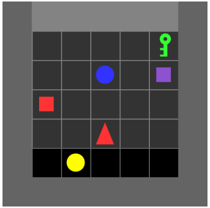
Effectively train RL agents to follow diverse instructions (**tasks**) in varied environments (**levels**)

Task



Reward Machine

Level



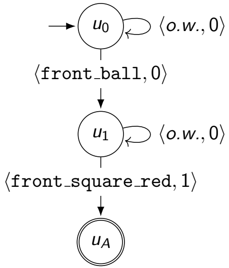
Agent: 

Training Generally Capable Agents

Goal

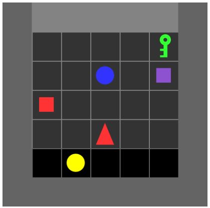
Effectively train RL agents to follow diverse instructions (**tasks**) in varied environments (**levels**)

Task



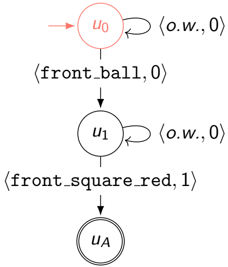
Reward Machine

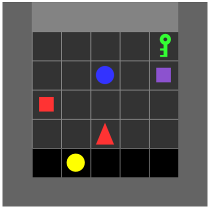
Level



Agent: ▲

Rollout



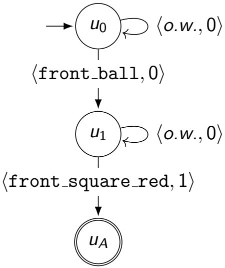


Training Generally Capable Agents

Goal

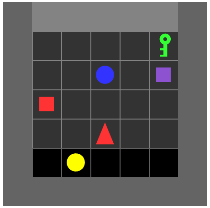
Effectively train RL agents to follow diverse instructions (**tasks**) in varied environments (**levels**)

Task



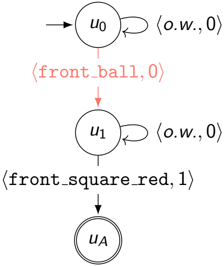
Reward Machine

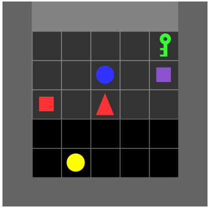
Level



Agent: 

Rollout



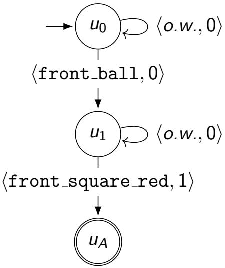


Training Generally Capable Agents

Goal

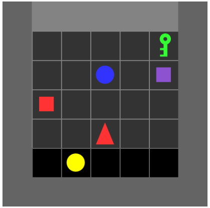
Effectively train RL agents to follow diverse instructions (**tasks**) in varied environments (**levels**)

Task



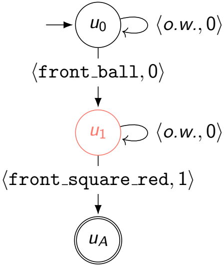
Reward Machine

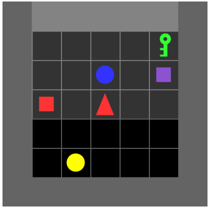
Level



Agent:

Rollout



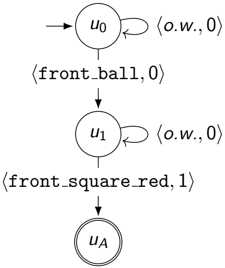


Training Generally Capable Agents

Goal

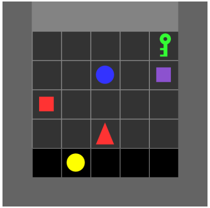
Effectively train RL agents to follow diverse instructions (**tasks**) in varied environments (**levels**)

Task



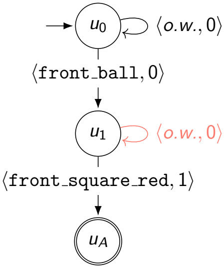
Reward Machine

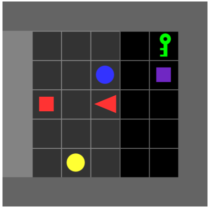
Level



Agent:

Rollout

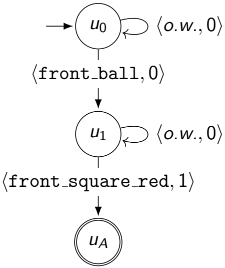




Goal

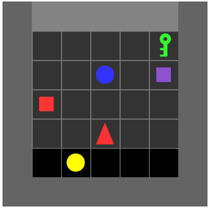
Effectively train RL agents to follow diverse instructions (**tasks**) in varied environments (**levels**)

Task



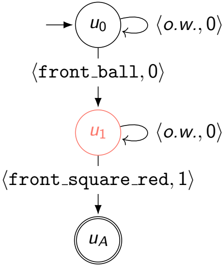
Reward Machine

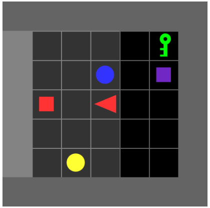
Level



Agent: ▲

Rollout



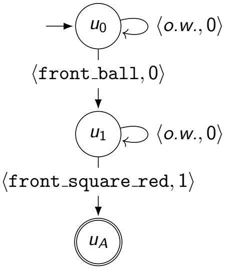


Training Generally Capable Agents

Goal

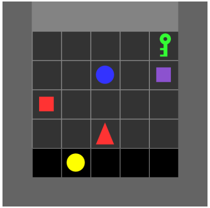
Effectively train RL agents to follow diverse instructions (**tasks**) in varied environments (**levels**)

Task



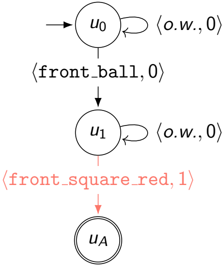
Reward Machine

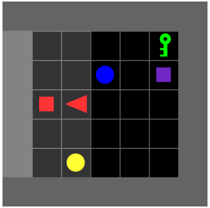
Level



Agent: ▲

Rollout



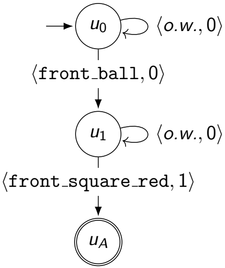


Training Generally Capable Agents

Goal

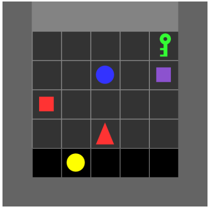
Effectively train RL agents to follow diverse instructions (**tasks**) in varied environments (**levels**)

Task



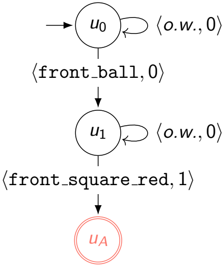
Reward Machine

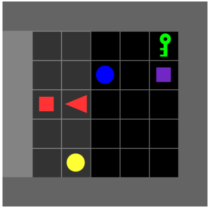
Level



Agent: 

Rollout





Task Specification with Formal Languages

Benefits w.r.t. Natural Language

- Precise, unambiguous semantics.
- Enable exact progress tracking through states.

Training via Domain Randomization (DR)

- Train on diverse, randomly sampled task-level pairs.
- **Challenge:** Uninformed sampling $\rightarrow$ many pairs are *unsolvable* or *too complex*.

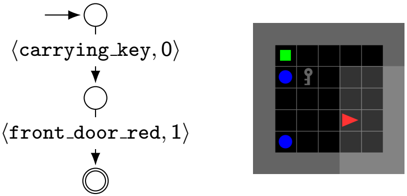
Task Specification with Formal Languages

Benefits w.r.t. Natural Language

- Precise, unambiguous semantics.
- Enable exact progress tracking through states.

Training via Domain Randomization (DR)

- Train on diverse, randomly sampled task-level pairs.
- **Challenge:** Uninformed sampling → many pairs are *unsolvable* or *too complex*.



Task requires red door, but none exists

How can we address this?
Generate adaptive curricula of solvable yet challenging task-level pairs

Unsupervised Environment Design (UED)

What is the goal?

- Generate *autocurricula*: training levels adapted to an agent's capabilities.

How does it work?

- Estimate the agent's *learning potential* on each level (e.g., regret).
- PLR$^{\perp}$ & ACCEL: Maintain a *buffer* of high-potential levels for training.

Jiang et al. (2021). "Prioritized Level Replay".
Jiang et al. (2021). "Replay-Guided Adversarial Environment Design".
Parker-Holder et al. (2022). "Evolving Curricula with Regret-Based Environment Design".

Unsupervised Environment Design (UED)

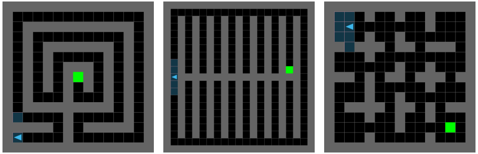
What is the goal?

- Generate *autocurricula*: training levels adapted to an agent's capabilities.

How does it work?

- Estimate the agent's *learning potential* on each level (e.g., regret).
- PLR$^{\perp}$ & ACCEL: Maintain a *buffer* of high-potential levels for training.

Gap: Prior UED only generates curricula over *levels* for a *fixed task*.

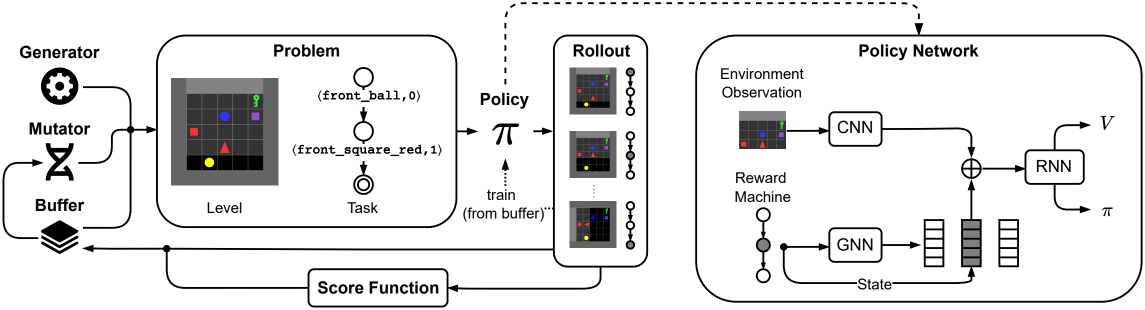


Diverse levels, fixed task (reach the green cell)

Jiang et al. (2021). "Prioritized Level Replay".
Jiang et al. (2021). "Replay-Guided Adversarial Environment Design".
Parker-Holder et al. (2022). "Evolving Curricula with Regret-Based Environment Design".

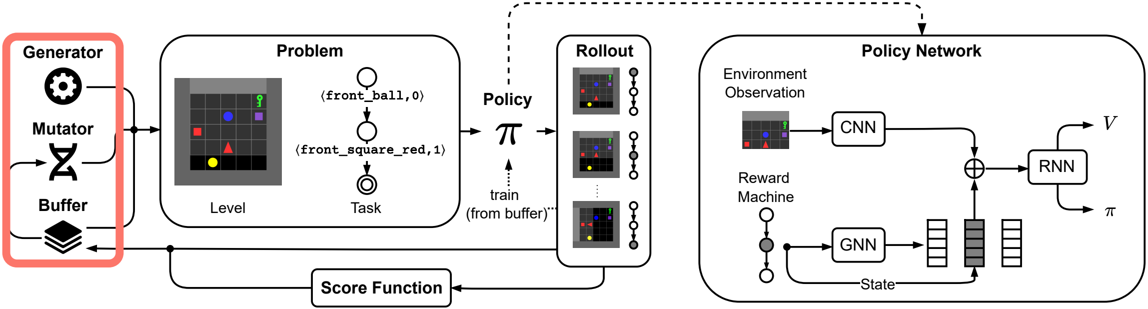
ATLAS: Aligning Tasks and Levels for Autocurricula of Specifications

Extend UED to generate **autocurricula** over **both** tasks and levels



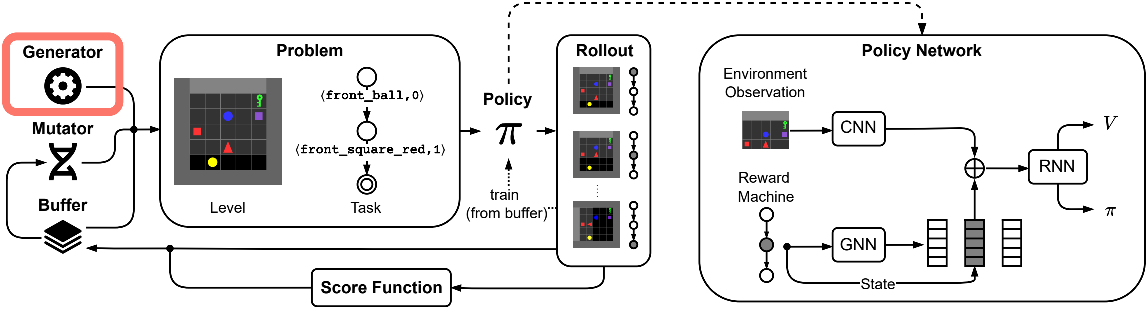
Co-design solvable yet challenging task-level pairs
by building on PLR$^{\perp}$ and ACCEL

Extend UED to generate **autocurricula** over **both** tasks and levels



At each step, **choose one of** Generator, Mutator (ACCEL), and Buffer
to determine the training task-level pairs

Extend UED to generate **autocurricula** over **both** tasks and levels

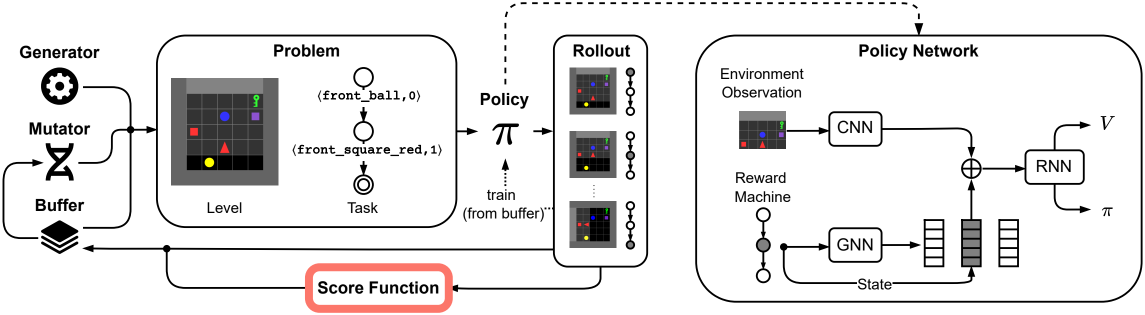


The **Generator** creates new task-level pairs via domain randomization (DR)

ATLAS: Aligning Tasks and Levels for Autocurricula of Specifications

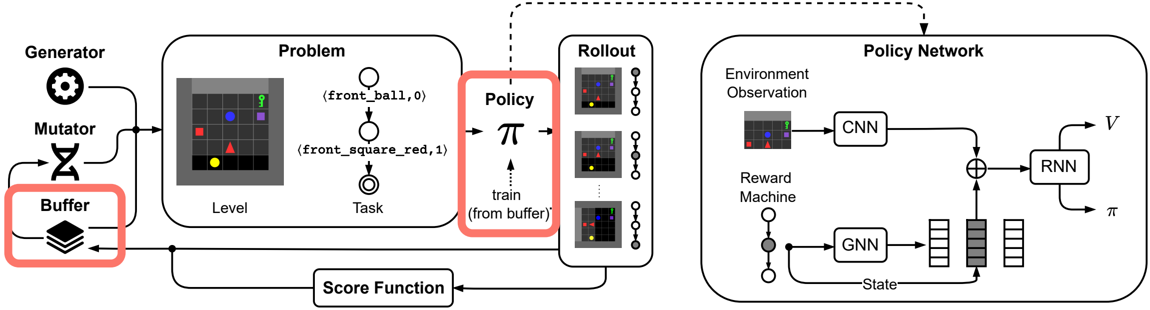
Overview

Extend UED to generate **autocurricula** over **both** tasks and levels



The **Score Function** estimates regret from **rollouts**: prioritize high-potential problems

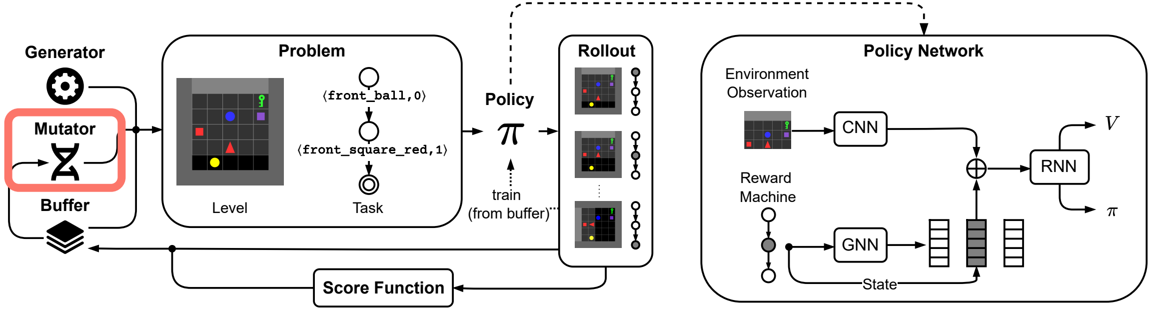
Extend UED to generate **autocurricula** over **both** tasks and levels



The **Buffer** stores task-level pairs prioritized by **score**

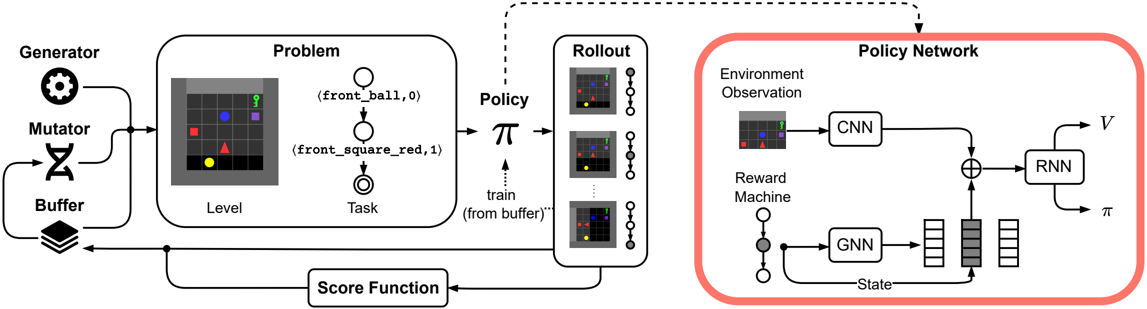
The **policy** is only trained on buffer-sourced task-level pairs (PLR$^{\perp}$)

Extend UED to generate **autocurricula** over **both** tasks and levels



The **Mutator** edits buffer task-level pairs (e.g., move an object, add RM state)

Extend UED to generate **autocurricula** over **both** tasks and levels



Policy Network: CNN (observations) + GNN (RM structure) for generalization

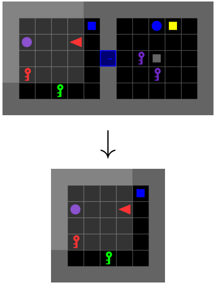
ATLAS: Aligning Tasks and Levels for Autocurricula of Specifications

Structure-Aware Task Mutations

Leverage RM structure to explore task-level neighborhoods

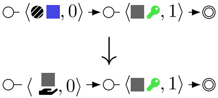
Level

Add/remove rooms, move/replace objects



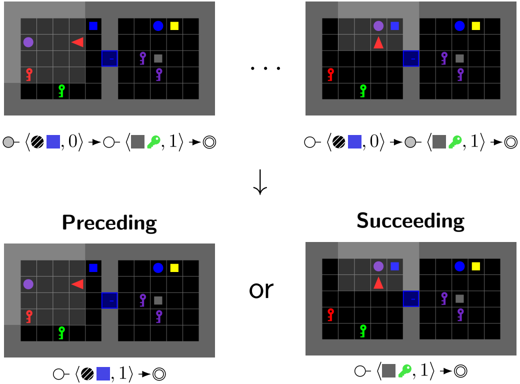
Task

Add/remove states, switch subgoals



Hindsight (Task + Level)

Extract sub-problems from rollouts

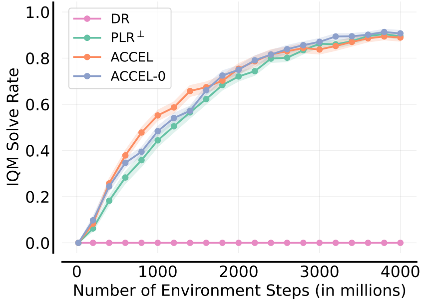


Contiguous symbols (e.g.,) indicate the objects must be observed next to each other. A hand under the object indicates the object must be carried. Striped patterns indicate "of any color".

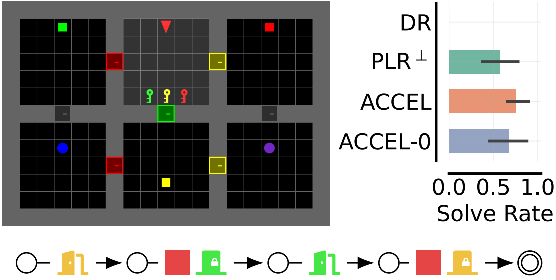
Results

Test Performance

ATLAS vastly outperforms Domain Randomization



Learning curves on hand-designed test set (150 problems)



Problem involving exploration + long-term planning

(a yellow door must be kept locked for the last subgoal)

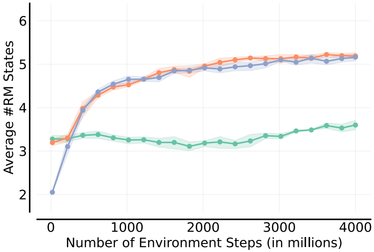
- **ATLAS (PLR$^\perp$, ACCEL):** ≈90% solve rate.
- **DR:** Near 0%.
- **ACCEL-0:** Strong performance from simple start

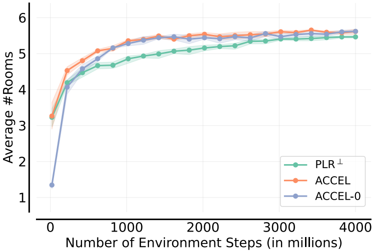
Why?

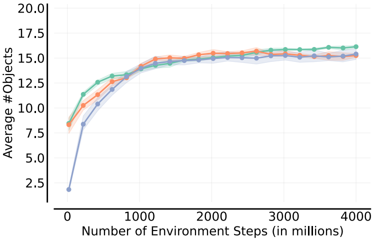
Only ≈2.7% of random pairs per DR batch are solvable!

Curriculum Emergence

Autocurricula emerge over BOTH tasks and levels





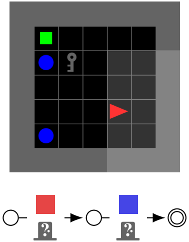


RM States	**Rooms**	**Objects**
$2 \rightarrow 5$ states	$1 \rightarrow 6$ rooms	$1 \rightarrow \approx 16$ objects

Generated Training Problems

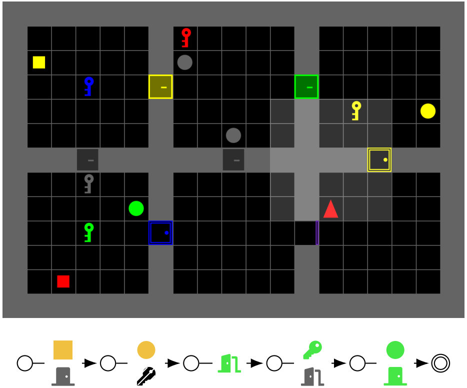
ATLAS generates solvable yet challenging problems

Domain Randomization



Random complexity
Mostly unsolvable

ACCEL-0



Curriculum from simple problems
Solvable + Challenging

Summary

Summary

Key Contributions

- **Joint task-level autocurricula:** Extended UED to co-design both tasks (RMs) and levels.
- **Structure-aware task mutations:** Leverage RM structure to explore task neighborhoods.
- **Strong empirical results:** $\approx$90% solve rate vs. DR's near 0% when solvable pairs are rare.

ATLAS enables generalization across diverse task-level pairs
by generating adaptive curricula of solvable yet challenging problems

Future Work

Exciting Directions

- **Richer domains:** Scale to more complex environments (e.g., Craftax).
- **Alternative scoring functions:** Beyond regret, exploit task structure.
- **Other task structures:** Hierarchies of RMs, linear temporal logic, programs, ...

Thank You!

Paper

https://arxiv.org/abs/2511.12706

Code

https://github.com/spike-imperial/atlas

Daniel Furelos-Blanco — d.furelos-blanco18@imperial.ac.uk