Hierarchies of Reward Machines

Daniel Furelos-Blanco[1], Mark Law[1,2], Anders Jonsson[3], Krysia Broda[1], and Alessandra Russo[1]

1 Imperial College London 2 ILASP Learning Logically 3 upf. Universitat Pompeu Fabra Barcelona

Motivation

- Learn at *multiple time scales* simultaneously.
- Learn with a rich *structure* of events and durations.

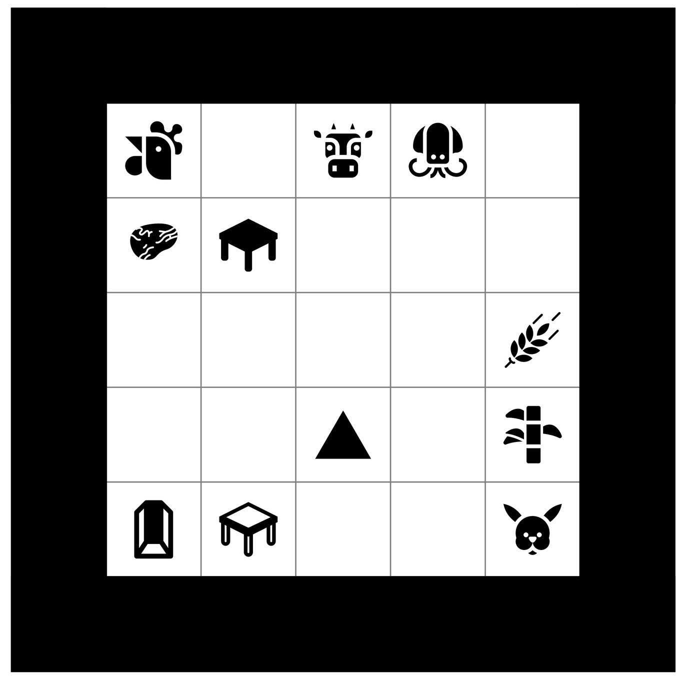
— **How?** —

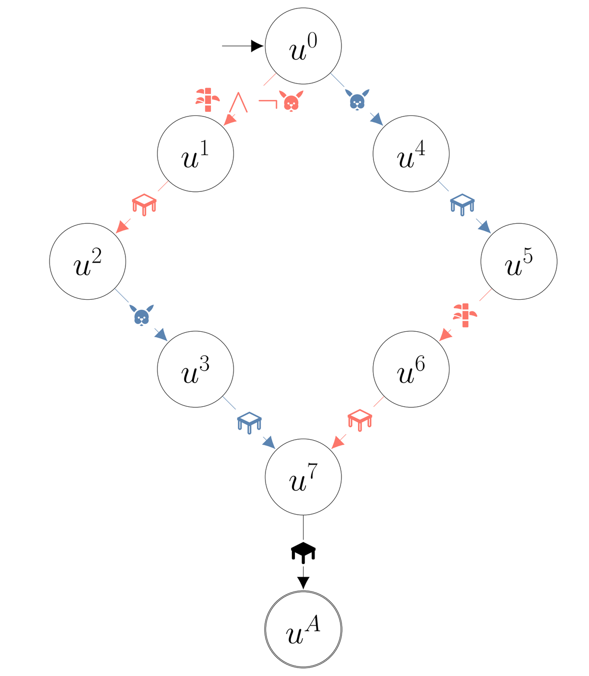
Using a type of finite-state machines called Reward Machines.

What is a Reward Machine (RM)?

A finite-state machine representation of a reward function using high-level propositional events $\mathcal{P}$.

Example: Observe then and then in any order, then .



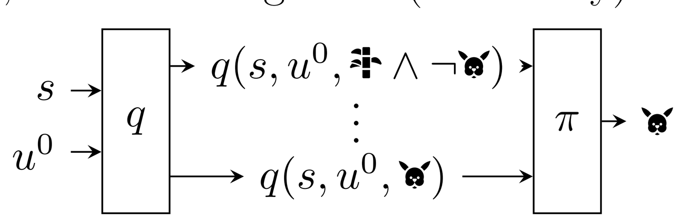


$\mathcal{P} = \{\ldots\}$ $\quad r(u, u') = \mathbb{1}[u \neq u^A \wedge u' = u^A]$

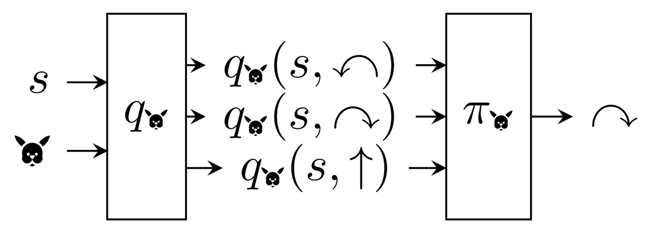
Policy Learning in RMs

Using the *options* framework for hierarchical reinforcement learning. There are two *decision levels*:

1. From an RM state, choose a subgoal to (eventually) reach u^A. Example:



2. Given a subgoal, choose an action to (eventually) satisfy it. Example:



Limitations of RMs

- Lack of *modularity*.
- *Hard to learn* when they contain more than a few states.

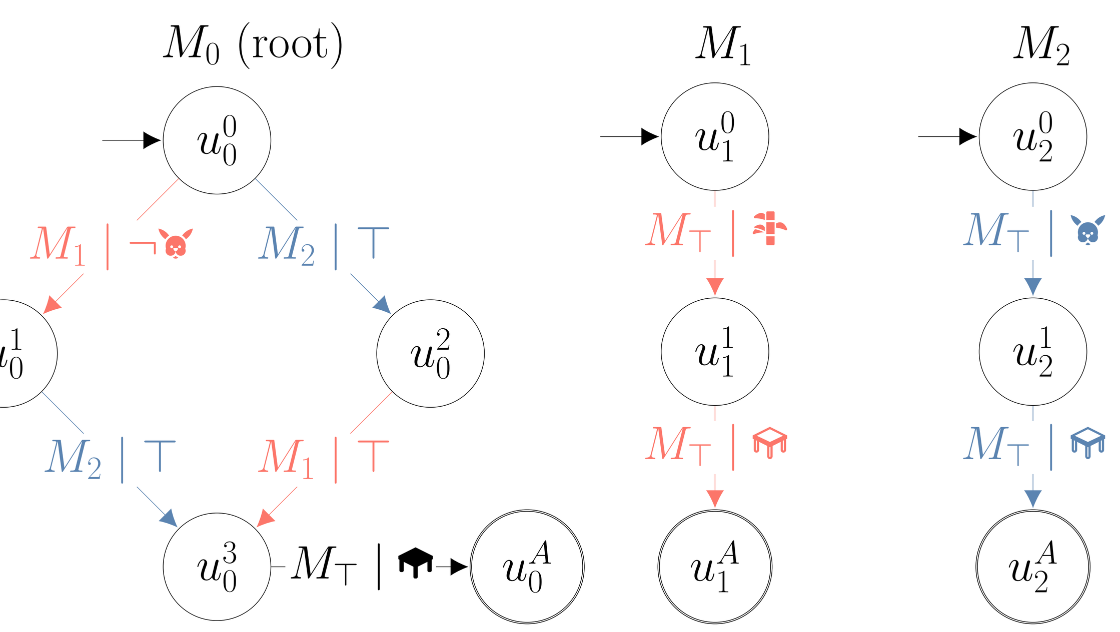
How to Address These?

Compose RMs into *hierarchies*.

Contributions

Hierarchies of Reward Machines (HRMs)

- Endow RMs with the ability to *call* each other.



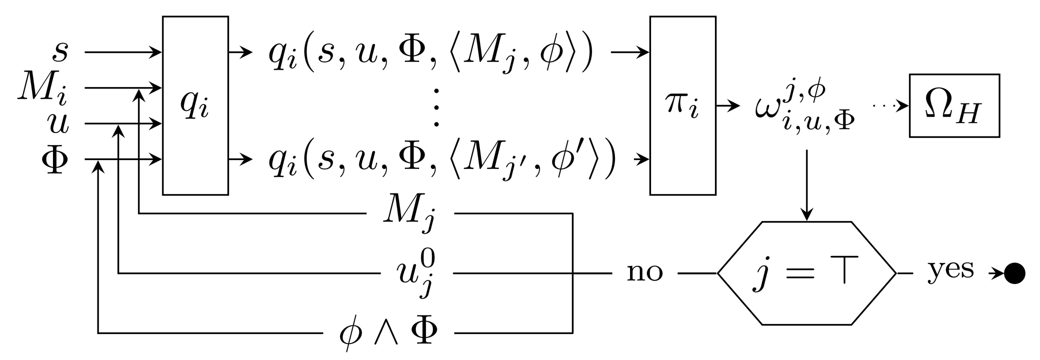
- Theory:
  1. Given an HRM, there exists an *equivalent flat* HRM.
  2. Given an HRM, an equivalent flat HRM *can* have *exponentially* more states and edges.

Hierarchically Compose + Exploit + Learn reward functions in the form of finite-state machines

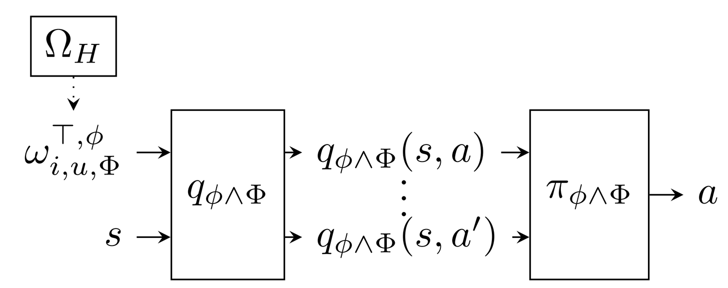
Policy Learning in HRMs

There are arbitrarily *many decision levels*, but still the *same two decision types*:

1. Iteratively choose subgoals top-down to reach the local u^A until a formula is selected.

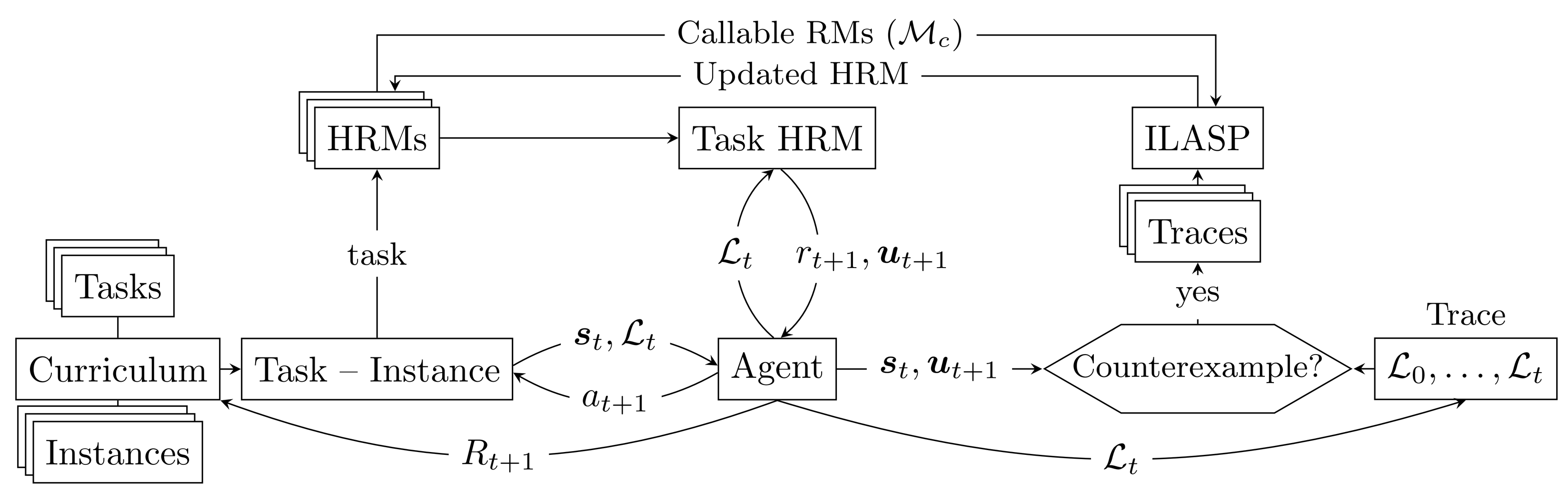


2. Choose an action to satisfy the selected formula.



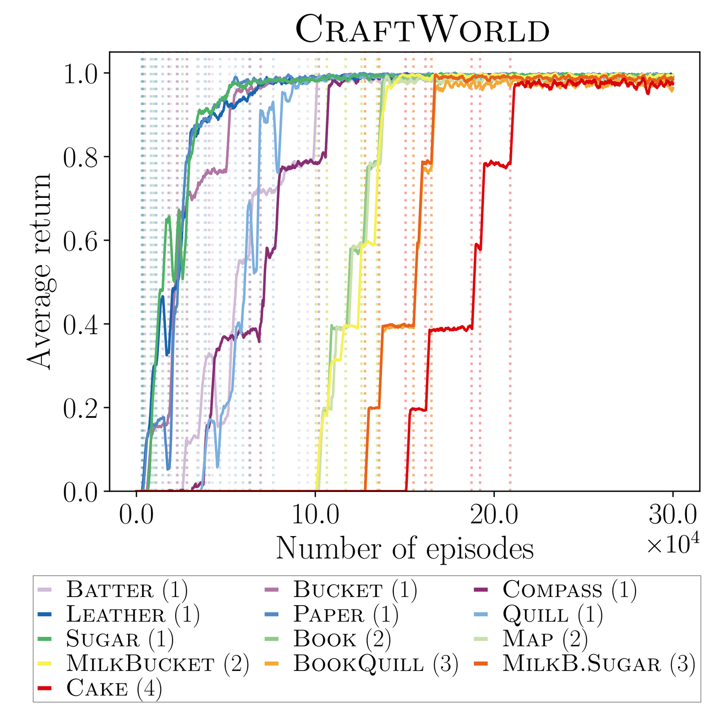
Learning HRMs from Traces

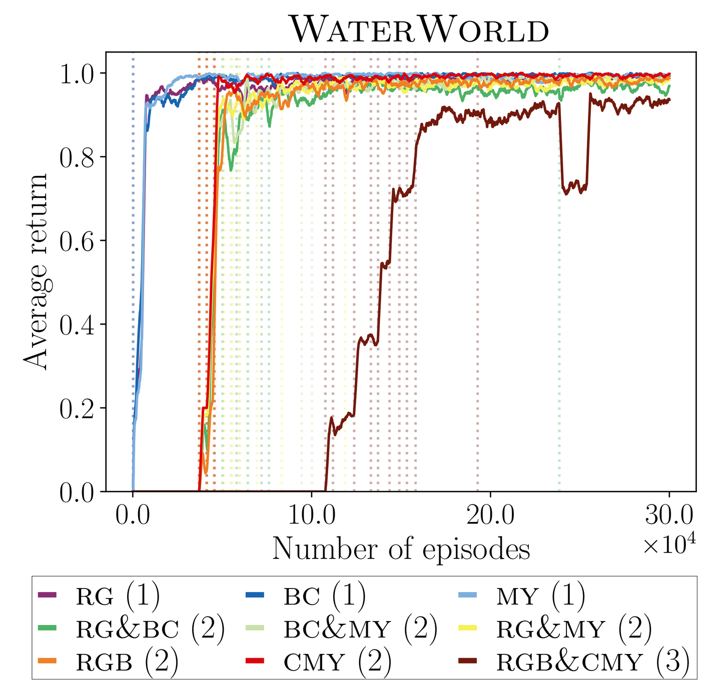
- Task-instance pairs are selected following a *curriculum learning* method.
- Each task is assigned to a *level*. Learning proceeds from lower to higher levels.
- HRMs are learned using the *ILASP* inductive logic programming system.



Results

Learning Non-Flat HRMs





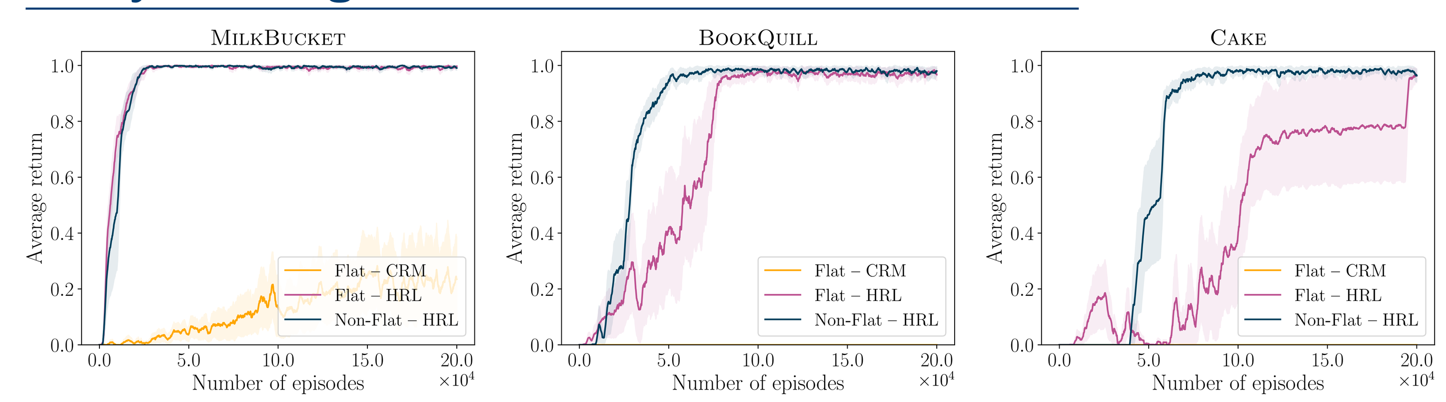
Ablations:

- A *restricted set of callable RMs* speeds up HRM learning by 5-7×.
- Using *options to explore* helps collect examples up to 128× faster.

Learning Flat HRMs

- We compare our method for learning a non-flat HRM against:
  1. Our method for learning a flat HRM.
  2. Existing RM learning methods (DeepSynth, JIRP, LRM) that label edges with proposition sets instead of formulas.
- Learning a non-flat HRM is more scalable than learning a flat HRM:
  - ⟹ Previously learned RMs are reused.
  - ⟹ The root may consist of fewer states and edges.
  - ⟹ Easier to learn!
- Abstraction through *formulas* is key in WaterWorld.

Policy Learning can be Faster in Non-Flat HRMs



Future Work

- Relax some *assumptions* (handcrafted propositions, fixed set of tasks).
- Learning in *non-episodic* settings.

Acknowledgments

Anders Jonsson is partially funded by TAILOR, AGAUR SGR and Spanish grant PID2019-108141GB-I00.

PAPER