Hierarchies of Reward Machines

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

[1]Imperial College London, UK [2]ILASP Limited, UK [3]Universitat Pompeu Fabra, Barcelona, Spain

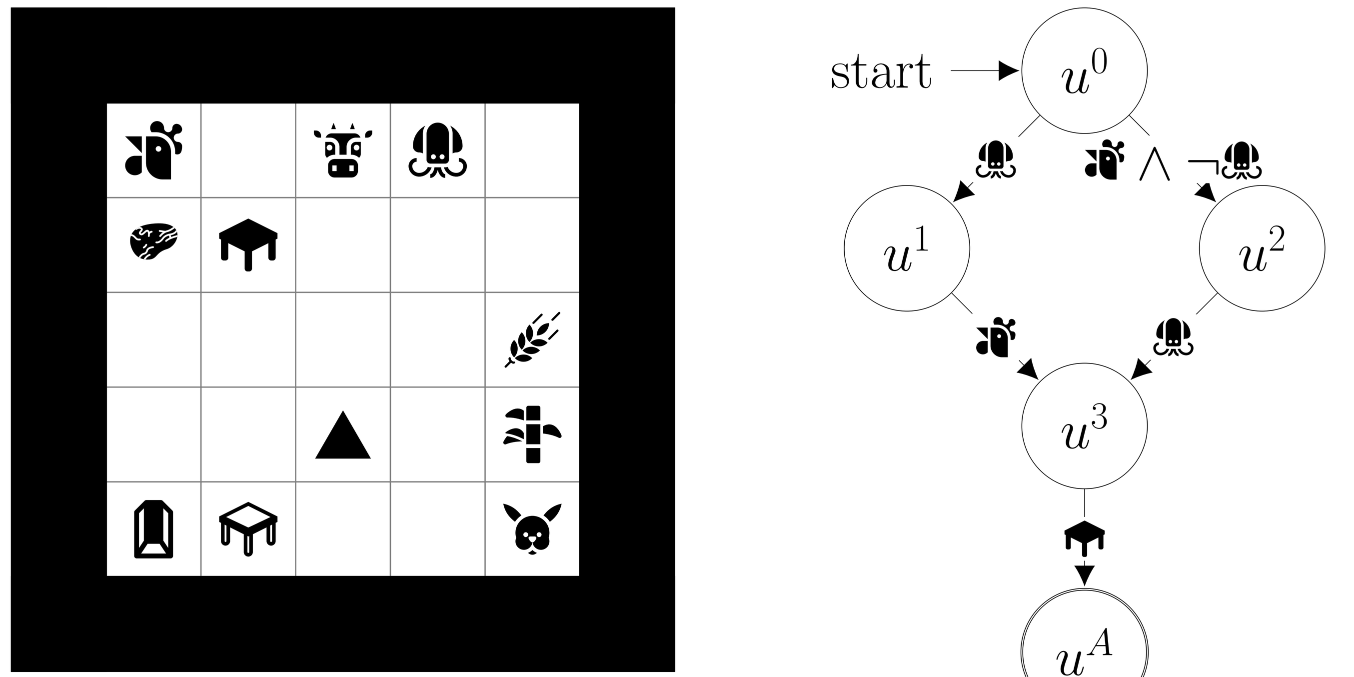
Motivation

- 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 [3].

What is a Reward Machine (RM)?



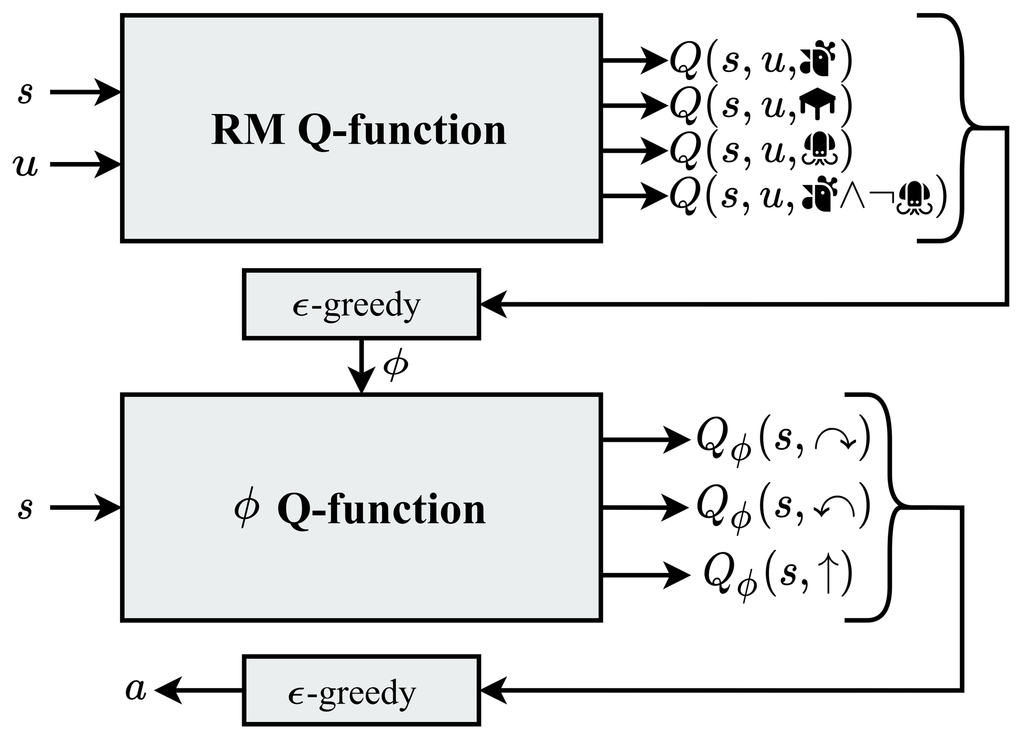
$\mathcal{P} = \{\ldots\}$

Task: Observe and in any order, then .

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

Policy Learning in RMs

Using the *options* framework [2] for hierarchical reinforcement learning.



Limitations of RMs

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

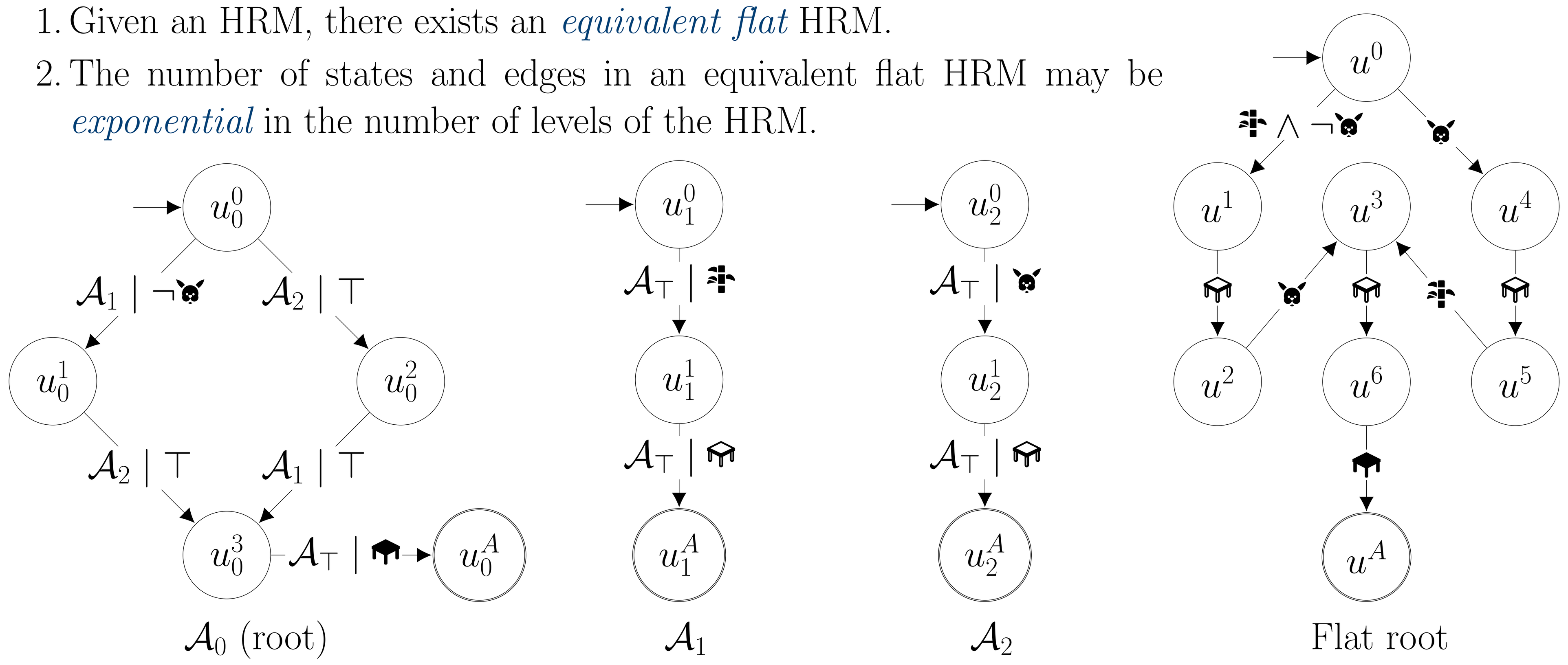
How to Address These?

Compose RMs into *hierarchies*.

Contributions

Hierarchies of Reward Machines (HRMs)

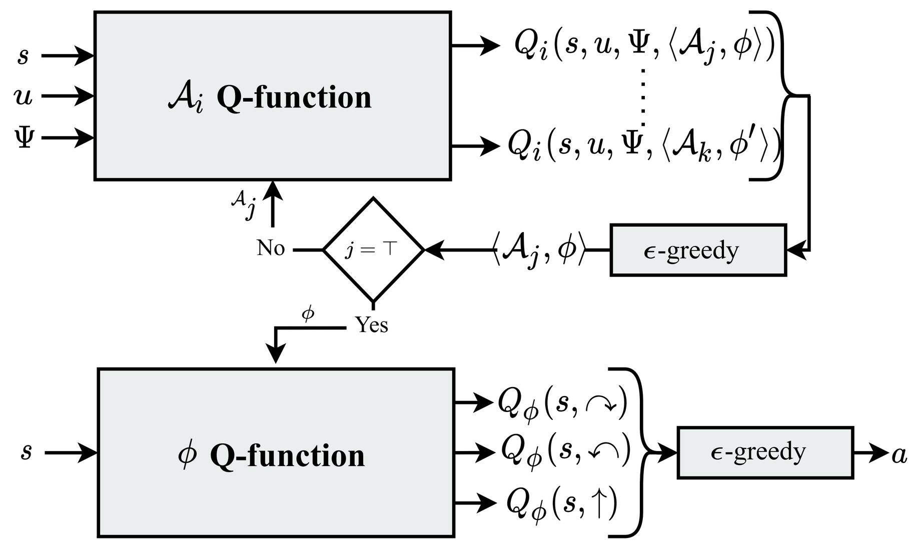
- Endue RMs with the ability to *call* each other.
- Theory:
  1. Given an HRM, there exists an *equivalent flat* HRM.
  2. The number of states and edges in an equivalent flat HRM may be *exponential* in the number of levels of the HRM.



Task	h	Description	Task	h	Description	Task	h	Description
Batter	1	(&) ;	Quill	1	(&) ;	BookQuill	3	Book & Quill
Bucket	1	;	Sugar	1	;	MilkB.Sugar	3	MilkBucket & Sugar
Compass	1	(&) ;	Book	2	(Paper & Leather) ;	Cake	4	Batter ; MilkB.Sugar ;
Leather	1	;	Map	2	(Paper & Compass) ;			
Paper	1	;	MilkBucket	2	Bucket ;			

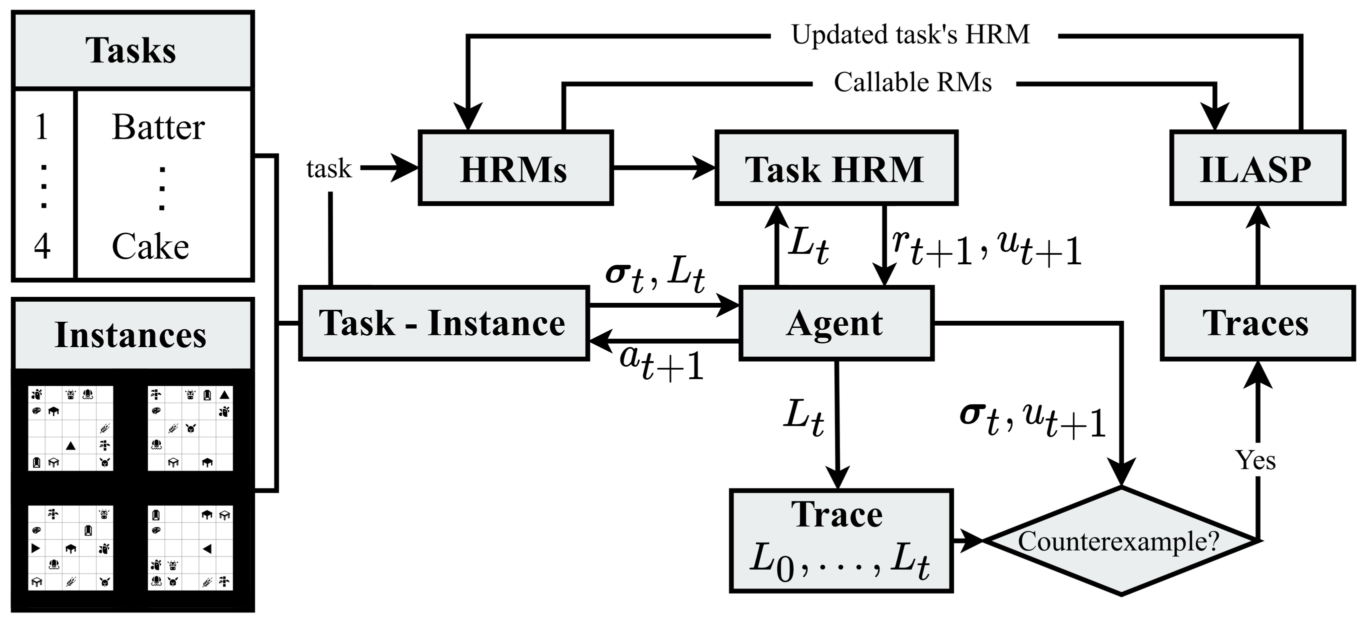
Learning Policies in HRMs

- Option types: *formula* options (associated with calls to $\mathcal{A}_\top$) and *call* options (other calls).



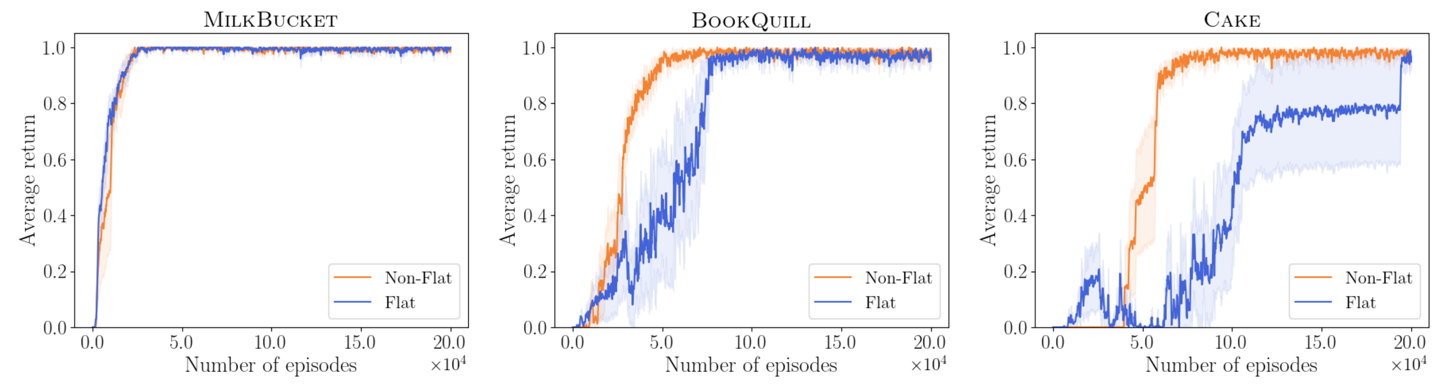
Interleaving Policy and HRM Learning

- Task-instance pairs are selected following a *curriculum learning* method.
- HRMs are learned using the *ILASP* inductive logic programming system [1].

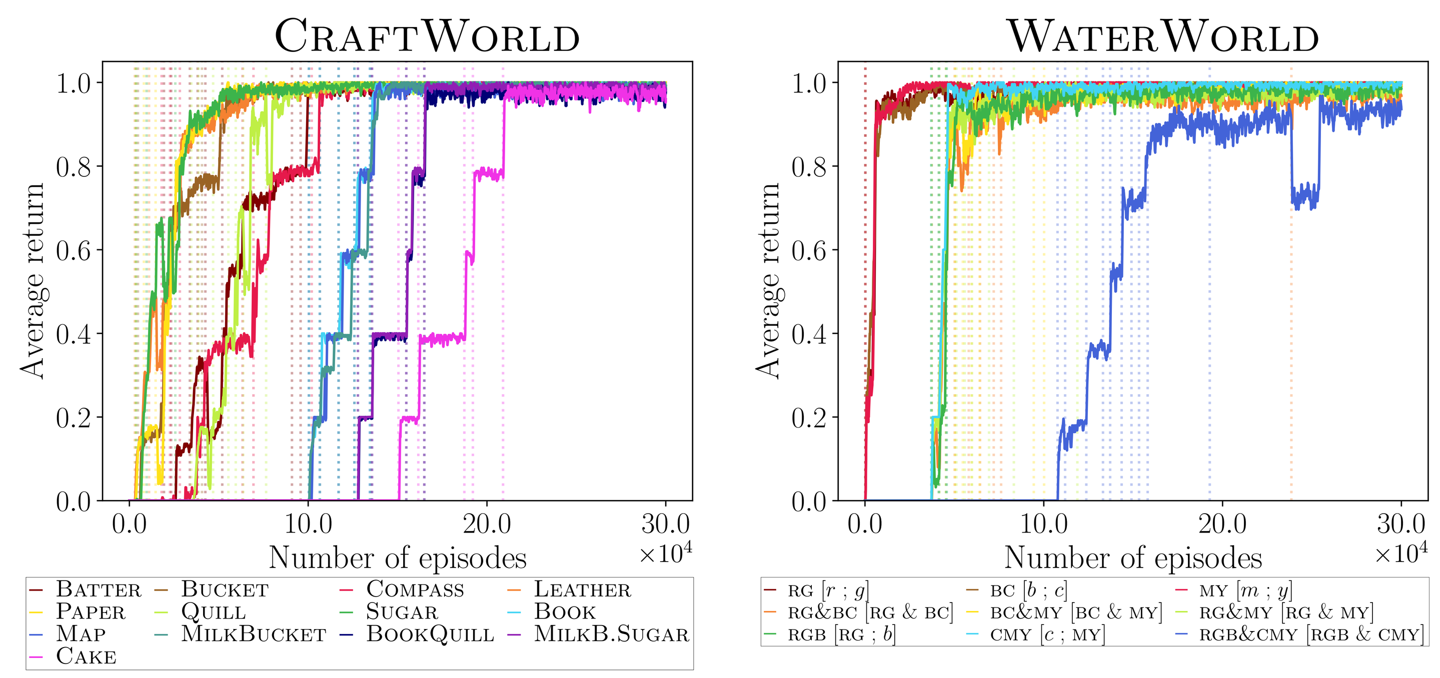


Results

Policy Learning can be Faster in Non-Flat HRMs



Learning Non-Flat HRMs



- A *restricted set of callable RMs* speeds up HRM learning by 5-7×.
- Using *options to explore* helps observing counterexamples faster (≈ 128× less episodes in some CW settings for the easiest tasks).

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 that label edges with proposition sets instead of formulas.
- Learning a non-flat HRM is more scalable than learning a flat HRM since it reuses previously learned RMs.
  - $\implies$ The root may consist of less states and edges.
  - $\implies$ Easier to learn!
- Abstraction through *formulas* is key in WaterWorld.

Future Work

Relax some *assumptions* (e.g., handcrafted propositions, the level of each task) and increase *generalization* across instances.

References

[1] M. Law, A. Russo, and K. Broda. The ILASP System for Learning Answer Set Programs, 2015.
[2] R. S. Sutton, D. Precup, and S. P. Singh. Between MDPs and Semi-MDPs: A Framework for Temporal Abstraction in Reinforcement Learning. Artif. Intell., 112(1-2):181–211, 1999.
[3] R. Toro Icarte, T. Q. Klassen, R. A. Valenzano, and S. A. McIlraith. Using Reward Machines for High-Level Task Specification and Decomposition in Reinforcement Learning. In ICML, 2018.