Logical Composition in Lifelong Reinforcement Learning

Geraud Nangue Tasse*, Steven James and Benjamin Rosman

University of the Witwatersrand, Johannesburg, South Africa

WITS

RAIL LAB

We propose a framework for lifelong learning that leverages zero-shot composition to solve new tasks that are expressible as logical combinations of learned ones.

Goal Oriented RL

We define an **extended value function (EVF)** that decouples the values for each absorbing state:

$$Q(s,g,a) = \bar{r}(s,g,a) + \int_S V^{\pi_g}(s')\rho_{(s,a)}(ds')$$

$$\bar{r}(s,g,a) = \begin{cases} N & \text{if } g \neq s \in G \\ r(s,a) & \text{otherwise} \end{cases}$$

Similar to UVFAs [3] but uses **extended rewards.**

Compositionality

Theorem 1: Let **M** be the set of tasks. Then **M** forms a Boolean algebra when equipped with the **or, and,** and **not** operators given by:

$$▮ \text{ or } ▀ = (S,A,p,r(▮ \text{ or } ▀)), \quad \text{where} \quad r(▮ \text{ or } ▀) = \max\{r(▮), r(▀)\}$$

$$▮ \text{ and } ▀ = (S,A,p,r(▮ \text{ and } ▀)), \quad \text{where} \quad r(▮ \text{ and } ▀) = \min\{r(▮), r(▀)\}$$

$$\text{not } ▀ = (S,A,p,r(\text{not } ▀)), \quad \text{where} \quad r(\text{not } ▀) = (r_{MAX} + r_{MIN}) - r(▀)$$

where, r_{MAX} and r_{MIN} are the reward functions for the maximum and minimum tasks.

Theorem 2: Let **Q** be the set of extended value functions. Then **Q** forms a Boolean algebra when equipped with the **or, and,** and **not** operators given by:

$$Q^*(▮) \text{ or } Q^*(▀) = \max\{Q^*(▮), Q^*(▀)\}$$

$$Q^*(▮) \text{ and } Q^*(▀) = \min\{Q^*(▮), Q^*(▀)\}$$

$$\text{not } Q^*(▀) = (Q^*_{MAX} + Q^*_{MIN}) - Q^*(▀)$$

where, Q^*_{MAX} and Q^*_{MIN} are the extended value functions for the maximum and minimum tasks.

Theorem 3: The task and extended value function spaces are homomorphic.

Lifelong RL With Composition

Algorithm 1: Lifelong RL with Composition

```
Input: Learning method 𝒜, task distribution 𝒟
𝒯 ← ∅
Q̄* ← ∅
while True do
    T, M ~ 𝒟
    ℰ𝒳𝒫 ← SumOfProducts(𝒯, T)
    if T = Evaluate(ℰ𝒳𝒫, 𝒯) then
        // Zero-shot transfer
        Q̄*_T ← Evaluate(ℰ𝒳𝒫, Q̄*)
    else
        // Learn and add to library
        Q̄*_T ← 𝒜(M)
        𝒯 ← 𝒯 ∪ T
        Q̄* ← Q̄* ∪ Q̄*_T
    end
    π*(s) ∈ arg max_{a∈𝒜} max_{g∈𝒢} Q̄*_T(s, g, a)
    Solve task M using π*(s) for all s ∈ 𝒮
end
```

Goals	M_0	M_1	M
g_0	0	0	0
g_1	0	1	1
g_2	1	0	1
g_3	1	1	0

Sum Of Products

- $M' \coloneqq (\neg M_0 \wedge M_1) \vee (M_0 \wedge \neg M_1)$
- $M = M'$? **Yes!**
- $Q^* = (\neg Q_0^* \wedge Q_1^*) \vee (Q_0^* \wedge \neg Q_1^*)$

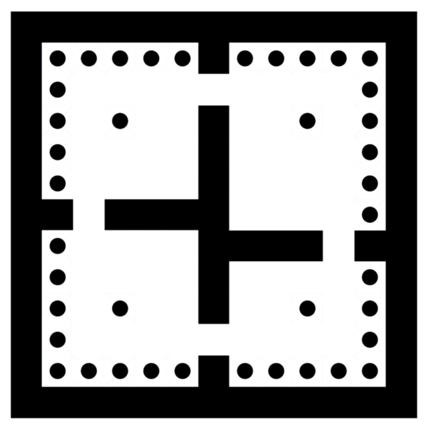
Goals	M_0	M_1	M
g_0	0	1	0
g_1	0	1	1
g_2	1	0	1
g_3	1	1	0

Sum Of Products

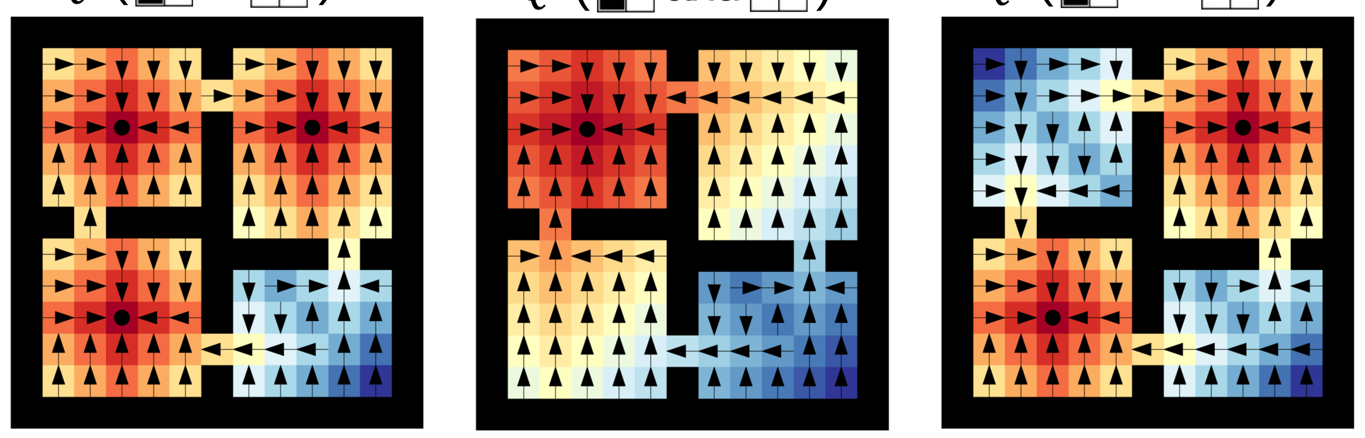
- $M' \coloneqq (\neg M_0 \wedge M_1) \vee (M_0 \wedge \neg M_1)$
- $M = M'$? **No!**
- Learn and add to library

Experiment: Zero-shot Composition

$\sim 10^{12}$ total tasks



$Q^*(▮ \text{ or } ▀)$ $Q^*(▮ \text{ and } ▀)$ $Q^*(▮ \text{ xor } ▀)$



Experiment: Lifelong RL With Composition