31.03.25

Recent Advances In HRL

Understanding the Need, Mechanisms, Applications, and Extensions

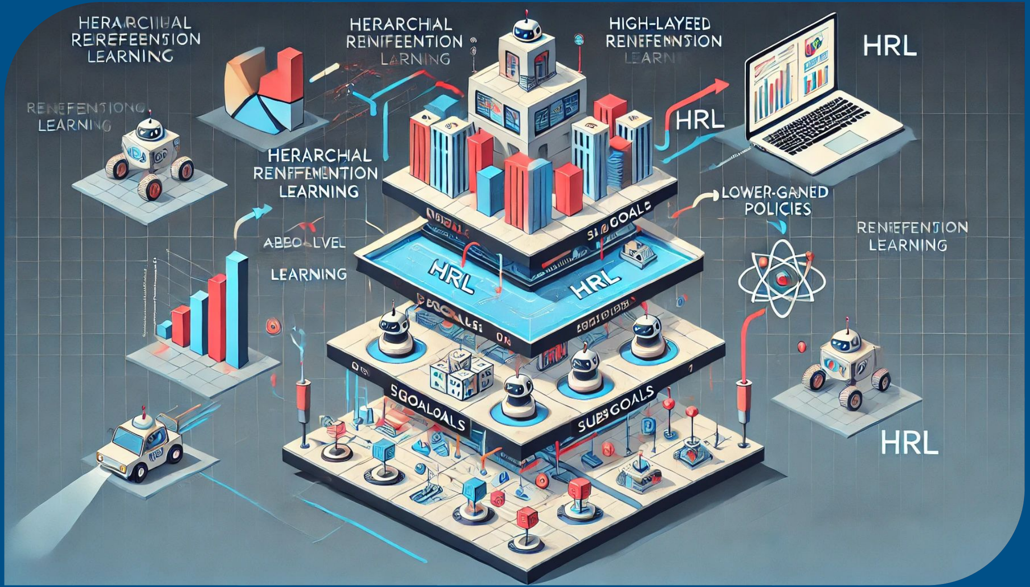


Image generated using AI for illustration purposes.

Rishabh Indoria
21F3001823

Hierarchical Reinforcement Learning (HRL) structures decision-making hierarchically by breaking tasks into sub-tasks, enabling more efficient learning and planning.

1. **Why do we need HRL?**
2. **What is HRL?**
3. **Real-World Applications**
4. **Recent Extensions**
5. **Open Research Challenges**

Long Horizon Tasks

Traditional RL struggles with multi-step decision-making due to sparse rewards.

Montezuma’s Revenge

Sample Inefficiency

RL agents require millions of interactions to learn optimal policies.

Space Invaders

Sparse Rewards & Exploration

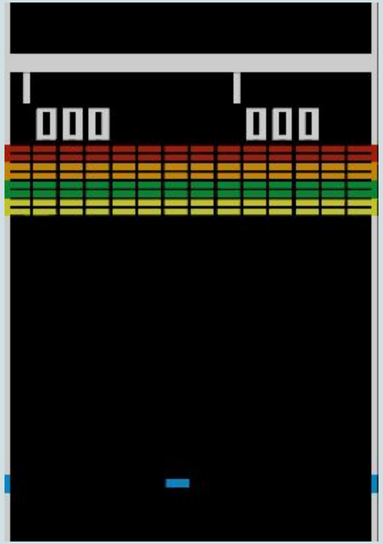
Traditional RL agents struggle with sparse rewards and inefficient exploration.

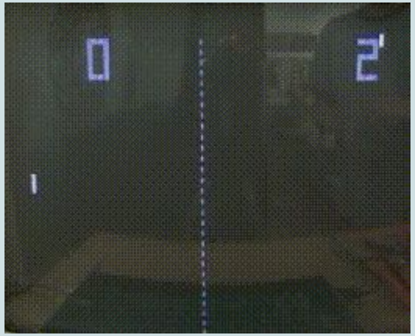
Pitfall!

Transfer Learning & Generalization

Skills learned in one task cannot be transferred to another.

Breakout vs Pong





Hierarchical Policy Structure

HRL consists of a high-level policy selecting sub-goals and low-level policies executing specific actions.

HRL reduces random exploration by structuring decisions into meaningful subtasks.

Montezuma's Revenge

Benefits of HRL

Long-Horizon Tasks

Efficient Subtask Execution: Break the task into hierarchical subgoals so the agent focuses on solving one manageable step at a time.

Sample Inefficiency

Reuse of Learned Skills: Store and reuse previously learned policies or skills to accelerate learning and reduce redundant exploration.

Sparse Rewards

Intermediate Rewards: Design intrinsic rewards or auxiliary tasks that provide feedback before reaching the final goal.

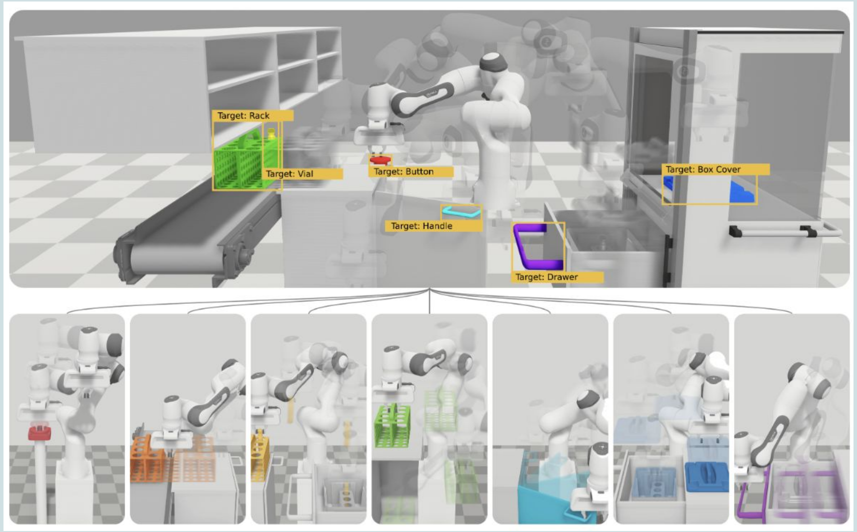
Generalization

Transferable Subskills: Train the agent to learn abstract skills that can apply across multiple environments or tasks.

1. Why do we need HRL?
2. What is HRL?
3. **Real-World Applications**
4. Recent Extensions
5. Open Research Challenges

HRL in Robotics

Used in grasping, manipulation, and locomotion.



Reference: https://arxiv.org/abs/2307.00125

HRL in Autonomous Driving

HRL structures driving tasks into navigation, lane-following, and obstacle avoidance.

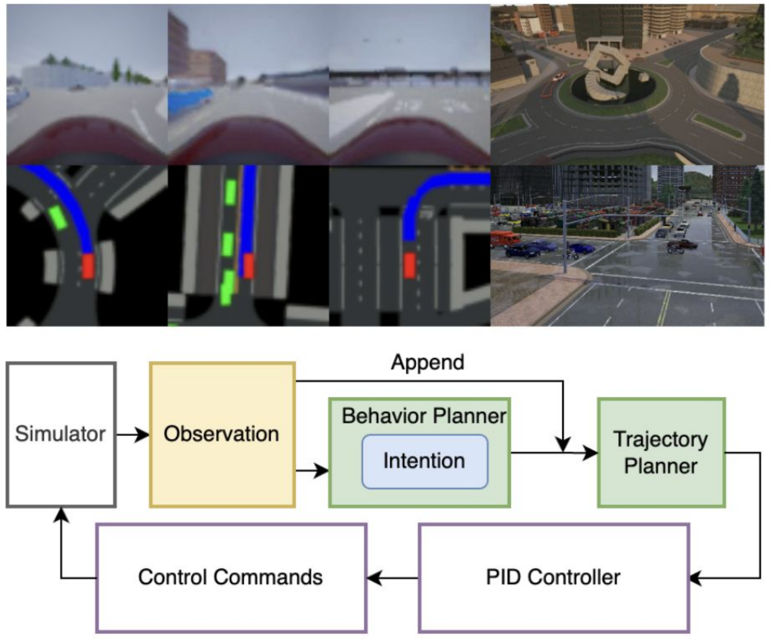
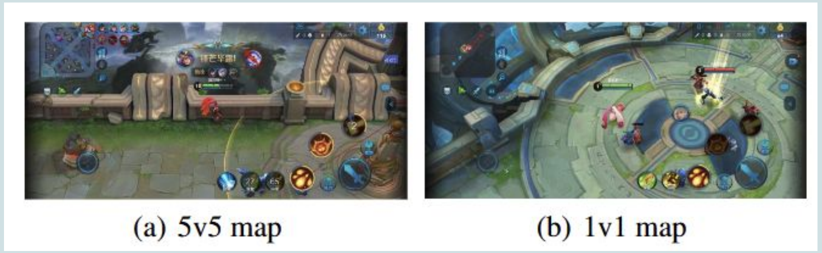


Figure 1. The architecture of the proposed hierarchical system

Reference: https://arxiv.org/abs/2306.15968

HRL in Game AI

Used in strategic decision-making.



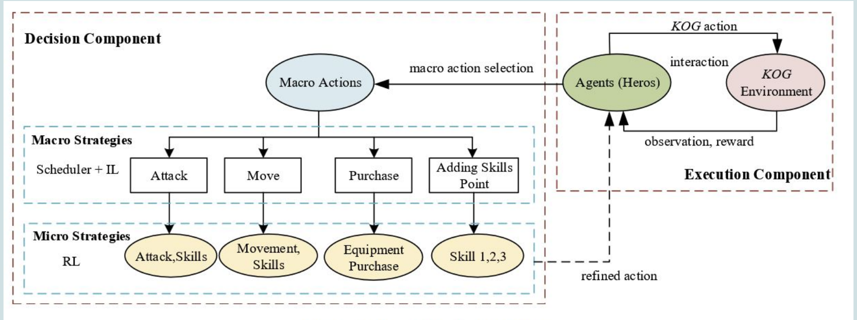


Figure 2: Hierarchical architecture

Reference: https://arxiv.org/abs/1901.08004

HRL in Healthcare

HRL structures treatment planning.

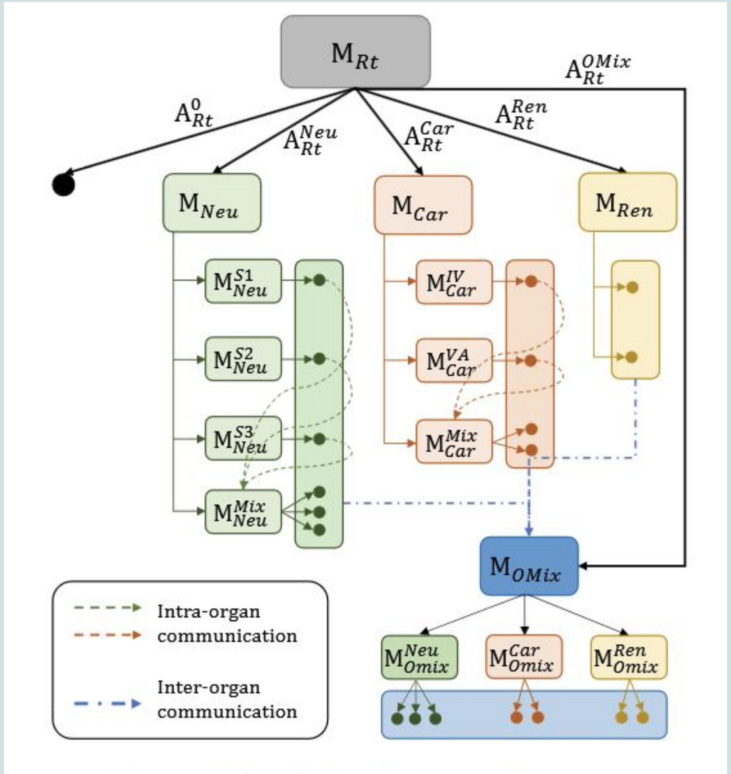
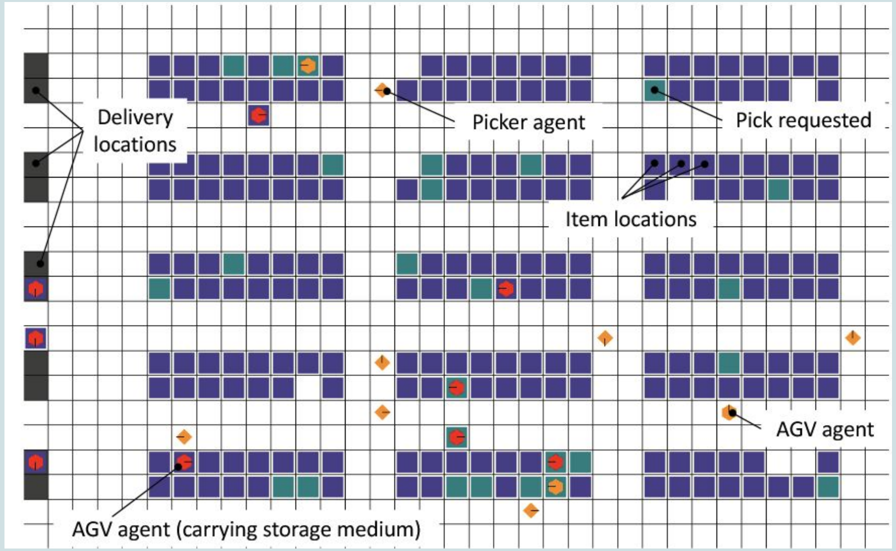


Figure 1: HMARL solution architecture.

Reference: https://arxiv.org/abs/2409.04224

HRL in Industrial Automation

Applied in warehouse logistics.



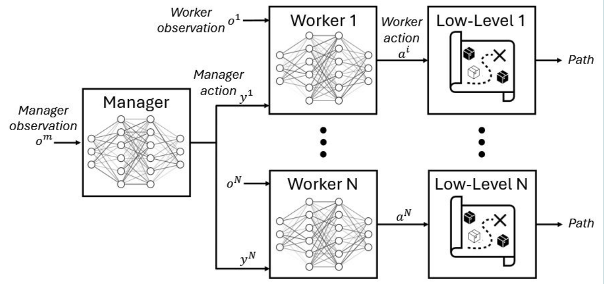


Fig. 2. Proposed 3-layer manager/worker agent hierarchy. A manager agent observes information about the warehouse state and orders, and assigns a task (target zone in warehouse) to each worker agent. Worker agents receive local observations about the warehouse and the assigned task from the manager, and select an item location from the assigned target zone. A low-level controller then navigates the worker to the selected item location.

Reference: https://arxiv.org/abs/2212.11498

1. Why do we need HRL?
2. What is HRL?
3. Real-World Applications
4. **Recent Extensions**
5. Open Research Challenges

Meta-HRL

Incorporates meta-learning for rapid policy adaptation.

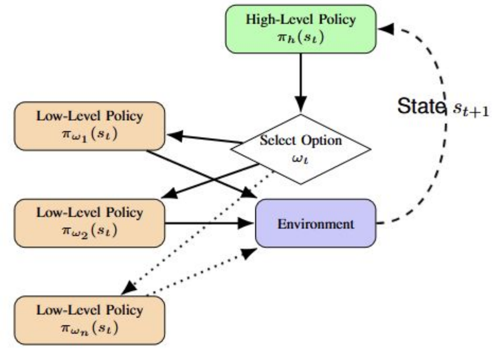

High-Level Policy $\pi_h(s_t)$
State s_{t+1}
Low-Level Policy $\pi_{\omega_1}(s_t)$
Select Option ω_t
Low-Level Policy $\pi_{\omega_2}(s_t)$
Environment
Low-Level Policy $\pi_{\omega_n}(s_t)$

Fig. 1. Architecture for Hierarchical Reinforcement Learning (HRL). After the high-level policy makes a choice, a low-level policy is triggered to engage with the environment. Based on input from the surroundings, the agent continuously modifies its state. The hierarchical flow of feedback and decision-making between high-level and low-level policies is shown by the dashed lines.

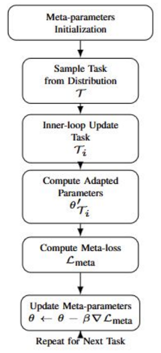

Meta-parameters Initialization
Sample Task from Distribution $\mathcal{T}$
Inner-loop Update Task $\mathcal{T}_i$
Compute Adapted Parameters $\theta'_{\mathcal{T}_i}$
Compute Meta-loss $\mathcal{L}_{\text{meta}}$
Update Meta-parameters $\theta \leftarrow \theta - \beta\nabla\mathcal{L}_{\text{meta}}$
Repeat for Next Task

Fig. 3. Meta-Learning Process Flowchart. The outer loop initializes and updates meta-parameters across tasks, while the inner loop performs task-specific adaptations using gradient descent. The meta-loss is computed based on adapted parameters to optimize the meta-parameters.

Reference:
https://arxiv.org/abs/2410.07921

Graph-Based Approach for Long-Horizon

Addresses the data inefficiency in large environments.

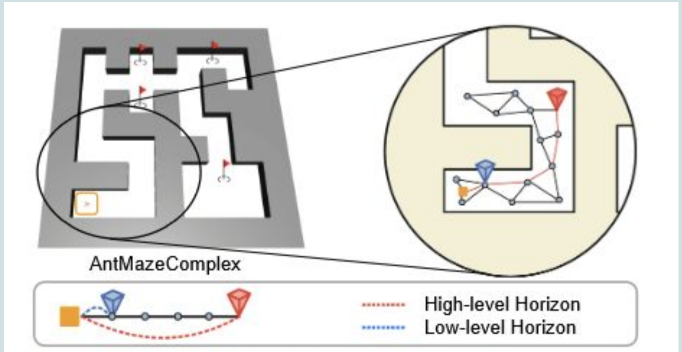


Figure 1: **DHRL:** By decoupling the horizons of both levels of the hierarchical network, DHRL not only solves long and sparse tasks but also significantly outperforms previous state-of-the-art algorithms.

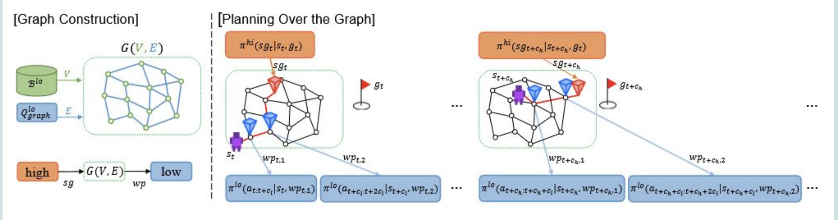


Figure 4: An overview of DHRL which includes the mid-level non-parametric policy between the high and low levels. The high-level (orange box) policy delivers subgoal $sg \in \mathcal{G}$ to the graph level (green box) and the graph instructs the low-level policy (blue box) to reach the waypoint $wp \in \mathcal{G}$. c_h represents how long each high-level action operates for. The low level is given $c_{l,i}$ steps to achieve the goal where $c_h \neq c_{l,i}$.

Reference: https://arxiv.org/abs/2210.05150

HRL in Complex 3D Environments

HRL has been extended to operate effectively in visually complex, partially observable 3D environments.

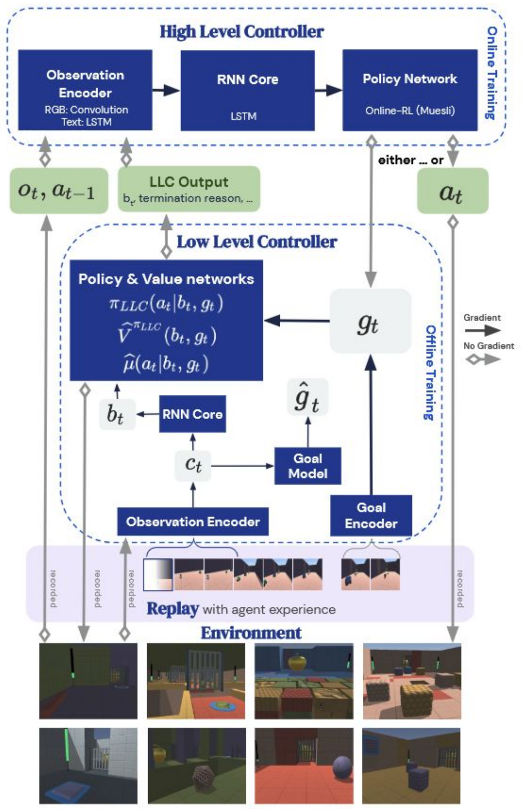


Figure 1 | H2O2 component diagram. The dotted boxes indicate how components (in blue) are trained. The arrows indicate information (inputs, outputs and gradients) passed between components.

Reference: https://arxiv.org/abs/2302.14451

Configurable Testbeds for HRL Evaluation

The development of scalable and flexible testing domains.

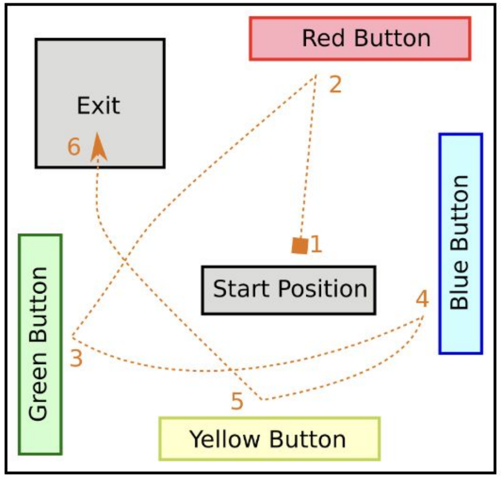


Figure 4.1: A bird's-eye view of the layout used for the ERD with embedded Button Puzzle (see Section 5).

Reference: https://arxiv.org/abs/1812.09521

Open Research Challenges

Learning Hierarchical Policies

Developing methods for agents to autonomously learn hierarchical structures without predefined sub-tasks or expert demonstrations is an ongoing challenge.

Subtask Discovery

Enabling agents to identify and create meaningful subtasks that contribute to overall task completion remains a complex problem.

Scalable Oversight

As AI models grow in complexity, developing scalable oversight mechanisms to provide reliable feedback and ensure alignment with desired behaviors is increasingly important.

Multi-Agent Learning

Extending HRL frameworks to effectively coordinate and learn in multi-agent settings poses challenges related to communication, cooperation, and scalability.