From End-to-end to Step-by-step: Learning to Abstract via Abductive Reinforcement Learning

Zilong Wang[1,2] , **Jiongda Wang**[2] , **Xiaoyong Chen**[2] , **Meng Wang**[2] , **Ming Ma**[3,4] , **Zhipeng Wang**[2] , **Zhenyu Zhou**[1,2] , **Tianming Yang**[3] , **Wang-Zhou Dai**[1,2]
[1]National Key Laboratory for Novel Software Technology, Nanjing University, China
[2]School of Intelligence Science and Technology, Nanjing University, China
[3]Institute of Neuroscience, State Key Laboratory of Brain Cognition and Brain-inspired Intelligence Technology, Center for Excellence in Brain Science and Intelligence Technology, Chinese Academy of Sciences, Shanghai, 200031, China
[4]University of Chinese Academy of Sciences, School of Future Technology, Beijing, 100049, China
{zilongwang, jiongdawang, xy.chen, meng_wang, jaycewang, zhenyuzhou}@smail.nju.edu.cn, {mam2022,tyang}@ion.ac.cn, daiwz@lamda.nju.edu.cn

Abstract

Abstraction is a critical technique in general problem-solving, allowing complex tasks to be decomposed into smaller, manageable sub-tasks. While traditional symbolic planning relies on predefined primitive symbols to construct structured abstractions, its reliance on formal representations limits applicability to real-world tasks. On the other hand, reinforcement learning excels at learning end-to-end policies directly from sensory inputs in unstructured environments but struggles with compositional generalization in complex tasks with delayed rewards. In this paper, we propose *Abductive Abstract Reinforcement Learning* (A2RL), a novel neuro-symbolic RL framework bridging the two paradigms based on Abductive Learning (ABL), enabling RL agents to learn abstractions directly from raw sensory inputs without predefined symbols. A2RL induces a finite state machine to represent high-level, *step-by-step* procedures, where each abstract state corresponds to a sub-algebra of the original Markov Decision Process (MDP). This approach not only bridges the gap between symbolic abstraction and sub-symbolic learning but also provides a natural mechanism for the emergence of new symbols. Experiments show that A2RL can mitigate the delayed reward problem and improve the generalization capability compared to traditional end-to-end RL methods.

1 Introduction

Abstraction is a powerful skill in problem-solving, which is able to reduce exponential problems to linear complexity [Korf, 1987]. It divides a difficult task into a set of subgoals, which are organized and solved *step-by-step* according to their dependencies. In traditional planning, a task abstraction is usually represented as a procedure and defined with formal languages [Unruh and Rosenbloom, 1989], while relying on predefined symbolic predicates and operators to model the abstract structures, which strongly limits their application in many real tasks having only sub-symbolic inputs.

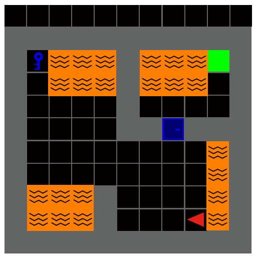


Figure 1: A `Minigrid` task. The target for ▶ is to reach 🟩. The top row is an inventory that displays the obtained items (such as 🔑).

In contrast, reinforcement learning (RL), especially Deep Reinforcement Learning (DRL), learns *end-to-end* policies and value functions directly from sensory inputs [Sutton and Barto, 2018; Mnih *et al.*, 2015], allowing it to adapt to diverse and unstructured environments without relying on predefined symbols or abstractions. This flexibility has led to (D)RL's success in a wide range of domains where abstractions might not be immediately available or easily defined. However, when the task is too complicated and the reward is delayed, this end-to-end feature of modern RL methods limits their compositional and sequential generalization capabilities [Arjona-Medina *et al.*, 2019; Ladosz *et al.*, 2022].

For instance, Figure 1 illustrates a task that presents challenges for many standard DRL methods. The environment is a grid world generated by `Minigrid` [Chevalier-Boisvert *et al.*, 2023], where an agent ▶ must navigate towards the goal 🟩 while avoiding lava blocks ▨. Additionally, the environment may feature a locked door 🟦, obstructing the path to the goal. To proceed, the agent might need to acquire a key 🔑 to unlock the door. Since the positive reward is only granted when the agent reaches the goal 🟩, actions such as obtaining 🔑 or unlocking 🟦 do not offer immediate feedback to the RL algorithm. As a result, without incorporating any inductive bias, learning a successful end-to-end policy in this environment becomes challenging.

This paper aims to bridge the gap between sub-symbolic RL and symbolic abstraction. We introduce the *Abductive Abstract Reinforcement Learning* (A2RL) framework, which employs an impasse-driven abstraction strategy [Unruh and Rosenbloom, 1989] to decompose a complex Markov Decision Process (MDP) into a series of sub-tasks, where each of them is modelled as a sub-MDP of the original problem. A2RL then constructs an *Abstract State Machine* (ASM) to formally represent the abstract procedure for solving these sub-tasks step-by-step. Finally, using the induced ASM structure, it trains end-to-end policies for each sub-MDP within the Abductive Learning (ABL) framework [Dai *et al.*, 2019; Zhou, 2019].

Another motivation of this paper is to explore the possibility of defining an "*open-universe* language that can *infer* the existence of objects from raw data (such as pixels, strings, etc.) that contain no explicit object references" [Russell, 2015]. We demonstrate that by transforming an end-to-end MDP into a step-by-step ASM, the transitions between abstract states naturally reveal new *symbols* from raw data, as shown in Figure 2. We conducted experiments on two sets of benchmark tasks, and the results showed that A2RL effectively learned the abstract structure, improving performance and easing the challenge of learning from delayed feedback. Additionally, a series of random map tests show that the policies learned by A2RL exhibit better generalization ability, as they can be easily reused and adapted to unseen tasks.

The rest of the paper is organized as follows. Section 2 covers the preliminaries, while Sections 3 and 4 formally define the *Abductive Abstract Reinforcement Learning* (A2RL) framework and explain its implementation. Section 5 presents the experiments and discussion, Section 6 reviews related work that inspired this study, and Section 7 concludes.

2 Preliminaries

Markov Decision Process. A Markov Decision Process (MDP) $\mathcal{M}$ is a tuple $\langle \mathcal{S}, \mathcal{A}, P, R, \gamma \rangle$, where $\mathcal{S}$ is state space; $\mathcal{A}$ is the action space; $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition probability function. $P(s'|s, a)$ is the probability of transitioning to state s' from state s after taking action a; $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function. $R(s, a, s')$ is the immediate reward received after taking action a in state s and reach s'; and $\gamma \in [0, 1]$ is the discount factor, which represents the difference in importance between future rewards and present rewards. The objective in an MDP is to find a policy $\pi : \mathcal{S} \to \mathcal{A}$, which maximizes the expected sum of discounted rewards over time, defined as the value function $V^{\pi}(s) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \mid s_0 = s\right]$, where $a_t = \pi(s_t)$.

State Abstraction. State abstraction is a process that groups a large amount of low-level *ground* states into a smaller set of high-level *abstract* states [Kersting *et al.*, 2004]. Formally, given a MDP with a raw state space $\mathcal{S}$, an *abstract state* $\sigma \subseteq \mathcal{S}$ is a set of ground states. All abstract states form a countable set $\Sigma = \{\sigma_0, \sigma_1, \sigma_2, \ldots\}$. An *abstraction function* ϕ is an indicator function, where $\phi(s) = j$ iff $s \in \sigma_j$.

Abductive Learning. Abductive Learning (ABL) [Zhou, 2019; Zhou and Huang, 2021] is a hybrid neuro-symbolic (NeSy) framework that combines machine learning and logical reasoning through a defined interface. The machine learning model processes raw environmental data and translates it into logical facts, which the reasoning model uses for inference. If inconsistencies arise—when the perceived facts fail to deduce correct outputs—a hybrid optimization process adjusts the flawed perceptions and uses these corrections to retrain the machine learning model [Dai *et al.*, 2019]. When the logical knowledge base is incomplete (e.g., missing rules to reason about the facts), the framework can also induce the missing rules while jointly training the perception model [Dai and Muggleton, 2021].

Previous work in ABL has focused primarily on classification problems, where machine learning models classify raw inputs and produce probabilistic monadic logical facts like "`0.9` :: `one(` `)`" [Dai *et al.*, 2019; Dai and Muggleton, 2021]. This paper extends this paradigm by modelling machine learning models as dyadic predicates, representing atomic procedures for constructing complex strategies for RL. Additionally, the reasoning model provides an *open-universe language* (i.e., the primitive symbols are unknown), allowing the agent to build a rule-based, step-by-step procedure from scratch. For example, a two-step policy could be expressed as:

$$\boldsymbol{\pi}^*(a, b) \leftarrow \boldsymbol{\pi}_{0\to 1}(a, c) \wedge \boldsymbol{\pi}_{1\to 2}(c, b)$$

where $\boldsymbol{\pi}^*/2$ and $\boldsymbol{\pi}_{i\to j}/2$ in bold font are dyadic predicates representing the state-transition procedures derived from policy functions π^* and $\pi_{i\to j}$; their arguments a, b, c are states in the raw state space $\mathcal{S}$. The above rule can be translated to natural language as "Applying policy π^* to transition from state a to state b can be divided into two steps: execute $\pi_{0\to 1}$ from a to c, and then apply $\pi_{1\to 2}$ from c to b".

3 Framework

In this section, we introduce the Abductive Abstract Reinforcement Learning (A2RL) framework, which is formalized based on the *Abstract State Machine* (ASM).

Definition 1 (Abstract State Machine)**.** *For a ground MDP $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P, R, \gamma \rangle$, a step-by-step abstraction for $\mathcal{M}$ is an Abstract State Machine $\Delta = \langle \Sigma, \mathcal{T}, P, R', \gamma \rangle$, where $\Sigma = \{\sigma \mid \sigma \subseteq \mathcal{S}\}$ is a countable set of abstract states; $\mathcal{T} = \{\tau_{i\to j} \mid \tau_{i\to j} = \mathbf{t}(\sigma_i) \cap \mathbf{s}(\sigma_j)\}$ is a countable set of abstract state transitions; $\mathbf{s}(\sigma)$ and $\mathbf{t}(\sigma)$ are the subsets of initial states and terminate states of σ, respectively; $R' = \{R\} \cup \{R_0^{\sigma}, R_1^{\sigma}, \ldots\}$ is a set of augmented reward functions, and each R_i^{σ} is associated with abstract state σ_i.*[1]

Figure 3a illustrates an example of MDP abstraction. The key idea is to segment an RL agent's trajectory into sequential steps. This divide-and-conquer strategy reflects human

[1]This paper uses Latin letters to denote *ground* states and actions, while *abstract* states and transitions are denoted by Greek letters.

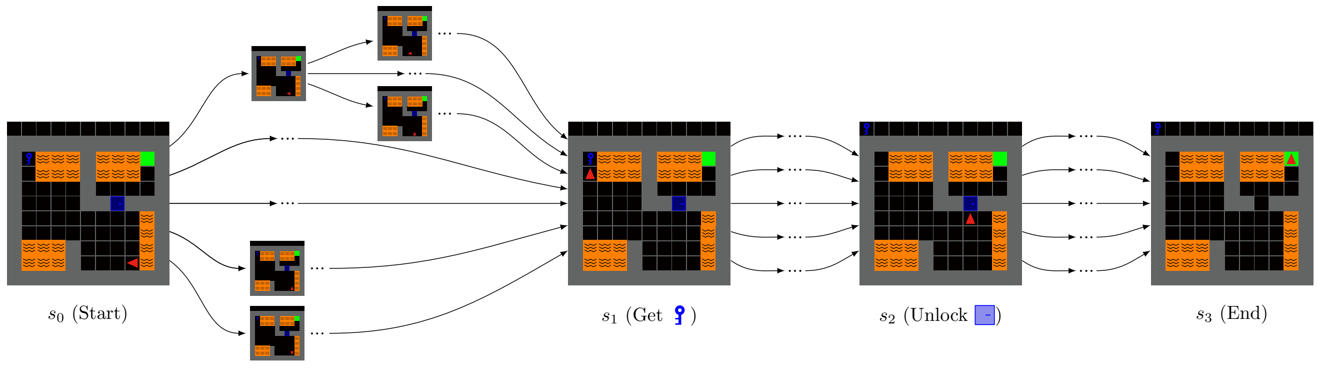
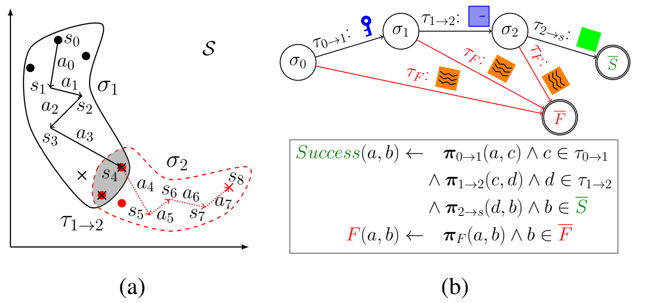


Figure 2: The key *ground* states in a typical `Minigrid` task. They segment all successful traces into 3 abstract steps, from which *symbols* such as “`key`” and “`door`” emerge naturally; their semantics, e.g., the visual representation of the symbols (`key` $\mapsto$ and `door` $\mapsto$), and the roles they play in the abstract state transitions could be automatically induced during learning.



$$
\begin{aligned}
Success(a,b) \leftarrow \quad & \boldsymbol{\pi}_{0\to 1}(a,c) \wedge c \in \tau_{0\to 1} \\
& \wedge \boldsymbol{\pi}_{1\to 2}(c,d) \wedge d \in \tau_{1\to 2} \\
& \wedge \boldsymbol{\pi}_{2\to s}(d,b) \wedge b \in \overline{S} \\
F(a,b) \leftarrow \quad & \boldsymbol{\pi}_F(a,b) \wedge b \in \overline{F}
\end{aligned}
$$

Figure 3: (a) A *ground* trace in $\mathcal{S}$ segmented into two *abstract* steps, illustrated as solid and dotted paths; $\bullet/\times$ in each colour denote the initial/terminate ground states in each step. (b) The induced ASM from the example in Figure 1 with its logic form.

decision-making. We employ automatic, intuitive processing (*System 1*) for simple tasks, relying on learned responses like “muscle memory”. For complex tasks, we engage deliberate, analytical processing (*System 2’*), decomposing the task into sub-tasks. If a sub-task is sufficiently simple, we revert to System 1; otherwise, we further decompose it [Kahneman, 2011]. A2RL adopts this approach and models abstracted steps as *sub-MDP*s solved by *atomic policy* functions.

Definition 2 (Sub-MDP and Atomic Policy)**.** *Given an ASM $\Delta = \langle \Sigma, T, P, R', \gamma \rangle$, the sub-MDP for an abstract state σ_i is a MDP $\mathcal{M}_i^\sigma = \langle \sigma_i, \mathcal{A}, P, R_i^\sigma, \gamma \rangle$, whose objective is to learn Atomic Policy functions $\pi : \sigma_i \to \mathcal{A}$ that navigates the agent from $\mathbf{s}(\sigma_\mathbf{i})$ to $\mathbf{t}(\sigma_\mathbf{i})$ while maximising the accumulated reward, which is calculated with the augmented reward function $R_i^\sigma : \sigma_i \times \mathcal{A} \to \mathbb{R}$.*

To ensure proper execution of the abstract plan, the transition $\tau_{i\to j}$ enforces the nose-to-tail constraint between $\mathcal{M}_i^\sigma$ and $\mathcal{M}_j^\sigma$, represented by logic rules or a graph model such as Figure 3b. If an abstract state σ_j serves as a terminal state, then $\tau_{*\to i} = \mathbf{s}(\sigma_i) = \mathbf{t}(\sigma_i)$, where “$*$” represents “any number in $\mathbb{N}$”. Since sub-MDP goals usually offer no rewards in the original MDP, a mental reward θ_i is introduced to define the augmented reward function R_i^σ. Formally,

$$
R_i^\sigma(s,a,s') = \begin{cases} \theta_i & \text{if } s' \in \mathbf{t}(\sigma_i), \\ R(s,a,s') & \text{otherwise} \end{cases}
$$

Note that there might be multiple j satisfying $\tau_{i\to j} \subseteq \mathbf{t}(\sigma_i)$, i.e., σ_i has more than one outgoing edge, so $\mathcal{M}_i^\sigma$ could have multiple atomic policies as its *options* [Bai *et al.*, 2016].

Next, we formally define the *Abductive Abstract Reinforcement Learning* (A2RL) task.

Definition 3 (Abductive Abstract Reinforcement Learning)**.** *Given a ground MDP $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P, R, \gamma \rangle$, the target for Abductive Abstract Reinforcement Learning is to:*

I. Discover a set of abstract states $\Sigma = \{\sigma_0, \ldots\}$ to construct the ASM $\Delta = \langle \Sigma, \mathcal{T}, P, R', \gamma \rangle$;

II. Learn a set of state abstraction functions Φ, where each $\phi_i : \mathcal{S} \to \{T, F\}$ such that $\phi_i(s) = T$ iff $s \in \sigma_i$;

III. Induce the abstract transition model $\mathcal{T}$;

IV. Solve the sub-MDPs derived from Δ by training atomic policy $\pi_{i\to j}$ for each state transition option $\tau_{i\to j}$.

As we can see, target I and II correspond to the logic reasoning model in the ABL framework, while target III and IV belong to the machine learning part. Specifically, target I requires the agent to infer the existence of logic *symbols* from raw data; target II learns a perception model mapping raw input to the logic symbols; target III induces a set of first-order logic rules to establish the relations between the discovered symbols; target IV learns a set of dyadic neural predicates to interface the logic rules with the sub-symbolic environment.

Note that A2RL only provides a *language*—first-order logic—to construct the ASM. Most of the symbols, including the abstract states and the primitive predicates for constructing the transition rules, are *undefined* and have to be discovered from raw data.

Object Discovery. Let’s rethink the question in [Russell, 2015]: can machine learning *infer the existence of objects from raw data that contain no explicit object references*? A2RL tries to answer it positively by defining *object* as an

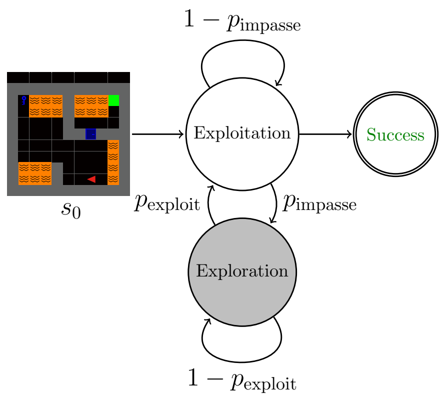


Figure 4: The impasse-driven exploration-exploitation of A2RL.

abstract state transition function that generalises across domains. For example, a `key` is an object that unlocks another object blocking the agent's way. In other words, the function of `key` is the most important thing for this concept, since it helps the agent change from a blocked state to unblocked, no matter it is a 🔑 in `Minigrid`, a bomb in The Legend of Zelda, or a rune stone in Tomb Raider. As shown in Figure 2 and 3b, one can easily learn the visual representation of an *object* and define the concept with logic rules from the state transitions of an ASM, which is induced from raw data.

Recursion. From an algebraic perspective, a sub-MDP forms a *subalgebra* of the original MDP, as its state space is a subset of the original and preserves the original MDP's algebraic structure. This definition enables the recursive division of sub-MDPs until it is trivial to solve. It not only simplifies the problem space but also maintains expressive completeness (with the aid of first-order logic language). Moreover, this recursive framework potentially offers a robust mathematical structure for further analysis and exploration.

4 Implementation

This section introduces a preliminary implementation for A2RL. Firstly, it tries to discover the novel abstract state σ_i using an impasse-driven strategy [Unruh and Rosenbloom, 1989]. Then it samples trajectories of σ_i by exploring the environment. Finally, abductive learning is applied to update the state abstraction function ϕ, the transition model $\mathcal{T}$, and learns the associated atomic policies $\pi_{i\to j}$.

4.1 Abstract State Discovery

As illustrated in Figure 2, a sample of ground MDP trajectories is required to discover the abstract state from raw inputs. In this paper, the data is collected through an impasse-driven exploitation-exploration procedure, as illustrated in Figure 4.

Exploitation. The exploitation is straightforward. Given an ASM $\Delta = \langle \Sigma, \mathcal{T}, P, R', \gamma \rangle$ and an input ground state s_t, the state abstraction function ϕ is called to identify its abstract state σ. A2RL then uses the abstract state transition $\mathcal{T}$ to calculate the optimal abstract plan Π^*—a path from σ_i to the target abstract state. Finally, the plan Π^* is executed step-by-step in the sub-symbolic environment by calling the atomic policies $\pi_{i\to j}$ in this path. The abstract plan Π^* always exists, since each $\pi_{i\to j}$ is the best policy that maximises the reward in sub-MDP $\mathcal{M}_i^\sigma$, and the transition rules $\tau_{i\to j}$ are correct.

Impasse. Nevertheless, when the current ASM $\Delta = \langle \Sigma, \mathcal{T}, P, R', \gamma \rangle$ represents an incomplete abstraction, the existing atomic policies may fail to construct an abstract plan capable of reaching the goal state from the initial state s_0. For example, in the `Minigrid` task illustrated in Figure 1, if the ASM contains only a single abstract state $\sigma_0 = \{s \mid$ door is open in $s\}$, the state abstraction function ϕ, implemented by a computer vision model, will classify any state $s \in \mathcal{S}$ as σ_0 since it is the sole option. Under the state transition rule $Success(a, b) \leftarrow \boldsymbol{\pi}_{0\to\text{goal}}(a, b) \wedge (b = \text{agent at } 🟩)$, the optimal plan Π^* includes only one end-to-end atomic policy, $\pi_{0\to\text{goal}}$. This policy, however, is unlikely to successfully navigate the agent to the goal in environments where the door is closed. In such cases, we say that Δ encounters an *impasse*.

In this paper, we employ the Drift Diffusion Model (DDM) to determine whether an agent is trapped in an impasse. The DDM is a widely used framework for simulating binary choice tasks—such as right or wrong judgments and left or right selections—in cognitive science and computational neuroscience [Ratcliff and McKoon, 2008; Gold and Shadlen, 2007]. It posits that decisions are made by progressively accumulating evidence until a certain threshold is reached. The DDM is typically formalized as a stochastic differential equation: $dx = v \cdot dt + \sigma \cdot dW_t$, where x represents the accumulated evidence, v is the drift rate, σ denotes the noise intensity, and W_t signifies standard Brownian motion.

We simply apply the DDM to identify impasses during agent-environment interactions. As the agent keeps operating without obtaining any reward or fails, the evidence x will accumulate and will trigger the agent to change from *exploitation* to *exploration* with the impasse probability $p_{\text{impasse}} \propto x$.

Exploration. The exploration strategy for A2RL is a random policy that randomly chooses a valid action in $\mathcal{A}$, and restarts if the agent fails (e.g., walks into 🟧 in `Minigrid` environment). During the rollout process, the state abstraction model ϕ is called to determine whether the sampled ground state s belongs to a known abstract state. We use DDM to make the choice, here the accumulating evidence $x = \max_i P(\phi(s) = i)$, and the strategy change from *exploration* to *exploitation* is triggered with probability $p_{\text{exploit}} \propto x$.

4.2 Abductive Learning

This section describes how to update the ASM Δ via abductive learning. In logic, *abduction* is the process of inferring the best explanation for a set of *observations*.

The impasse-driven exploration and exploitation rollout generates numerous ground trajectories as observations. Each trajectory can be explained as a step-by-step procedure, segmented into *exploration* and *exploitation* phases. The *exploitation* phases are further divided based on their corresponding abstract states. For the *exploration* segments, a clustering model is applied to the ground states, using raw

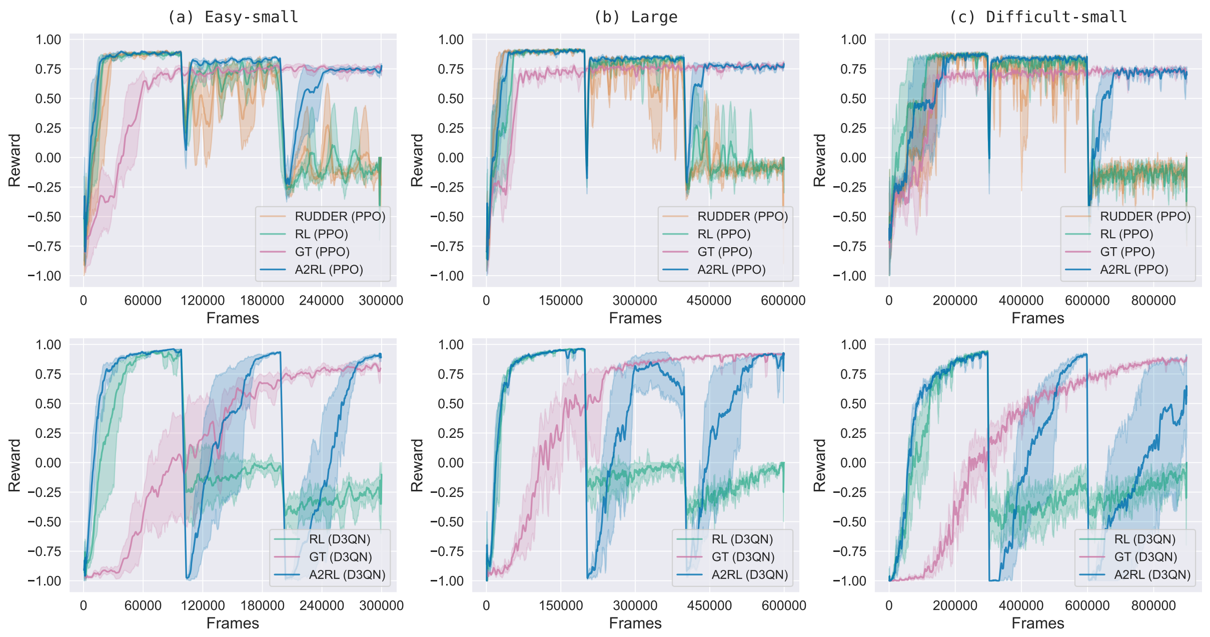


Figure 5: Training reward curve across three distinct maps. For each environment, the total training steps are set to 300k, 600k, and 900k, respectively. Note that both A2RL and RL curves are segmented into three distinct phases, reflecting the curriculum change. Each phase encompasses one-third of the total training steps.

images and neighbouring abstract states to group them and define a new abstract state, σ_{new}.

Then, the trajectory segments of σ_{new} are used for constructing $\mathbf{s}(\sigma_{\text{new}})$, $\mathbf{t}(\sigma_{\text{new}})$ and $\tau_{\text{new}\to\text{old}}$, where σ_{old} is a known abstract state in Σ. After adding a mental reward θ_{new}, a sub-MDP $\mathcal{M}^{\sigma}_{\text{new}}$ is created, from which the new atomic policies are trained. Meanwhile, the new state abstraction function ϕ_{new} is trained using the visited ground states. Finally, The elements Δ is updated as follows:

$$\begin{aligned}\Sigma &\leftarrow \Sigma \cup \{\sigma_{\text{new}}\}\\ \mathcal{T} &\leftarrow \mathcal{T} \cup \{\tau_{\text{new}\to\text{old}}\}\\ \Phi &\leftarrow \Phi \cup \{\phi_{\text{new}}\}\\ R' &\leftarrow R' \cup \{R^{\sigma}_{\text{new}}\}\end{aligned}$$

5 Experiment

Experiments[2] are designed to answer the following questions: **1:** Is A2RL able to infer the existence of abstract states from raw data? **2:** Does ASM improve the performance of RL in challenging environments(e.g., those with delayed rewards or sparse reward signals)? **3:** Does A2RL learn abstraction and atomic policies that generalize in unseen tasks?

5.1 Experimental Setup

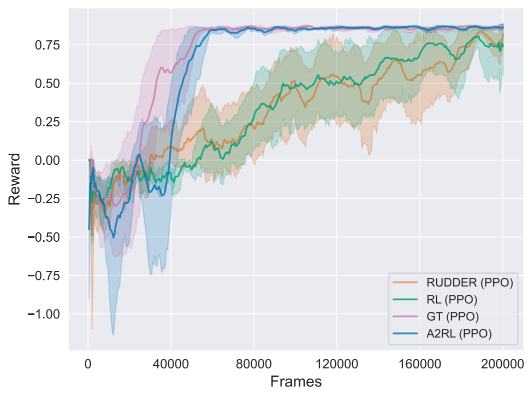
Environments. Experiments are conducted in two types of benchmark environments having delayed reward feedback: (1) `Minigrid` [Chevalier-Boisvert *et al.*, 2023]: Minigrid, implemented in Gymnasium [Towers *et al.*, 2024], is a suite of easily configurable grid world environments specifically designed for RL research. In `Minigrid`, we generated maps with different sizes: `Minigrid-small` and `Minigrid-large`. We further introduced two difficulty levels within the `Minigrid-small` environment: `Easy-small` and `Difficult-small`. (2) `Taxi` [Dietterich, 2000]: In `Taxi`, the taxi must pick up the passenger, drive to the destination, and drop them off to end the episode.

The input states contain solely raw pixels. This design choice demonstrates our method's capability to extract symbolic representations directly from raw inputs, aligning more closely with the requirements of real-world tasks.

Policy Learning. We employed two end-to-end DRL algorithms as base models for learning policy functions: (1) **PPO2** [Schulman *et al.*, 2017]: PPO2 is a model-free policy gradient method, the crux of which lies in Proximal Policy Optimization. Utilizing importance sampling, PPO2 effectively constrains the divergence between the old and new policies. (2) **D3QN** [Wang *et al.*, 2016; Hasselt *et al.*, 2016]: D3QN is a RL algorithm that amalgamates the strengths of the Dueling Network Architecture and Double Q-Learning, aiming to learn and evaluate Q-values more accurately.

Baselines. We compare A2RL with the following baselines: (1) **GT (Ground Truth)**: utilizes a handcrafted ASM with predefined symbols to set a performance upper-bound for A2RL. Its precise subtask decomposition means GT can be regarded as a Hierarchical Reinforcement Learning (HRL) approach [Levy *et al.*, 2019]. (2) **RUDDER**: A typical RL approach designed for tackling the delayed reward issue[Arjona-Medina *et al.*, 2019]. RUDDER was implemented on top of the TRPO-based policy gradient method PPO; therefore, it was only compared with the PPO baselines;

[2]The code is available at https://github.com/sporeking/A2RL.



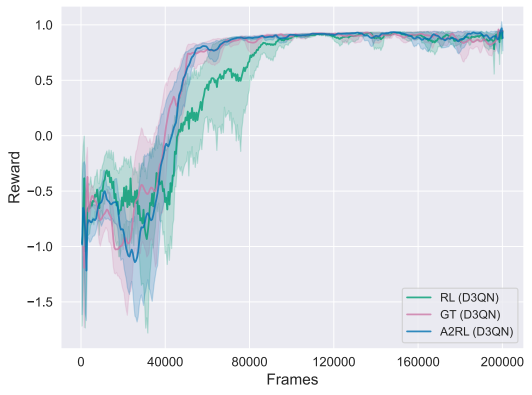


Figure 6: Training reward curves on Taxi environment.

(3) **RL**: Vanilla end-to-end DRL methods, without the assistance of symbolic reasoning.

Tasks. In Minigrid, three levels of progressively increasing difficulty were designed. On the one hand, the tasks in Minigrid are relatively complex, making it difficult for pure RL methods to converge; on the other hand, we can evaluate the effectiveness of our method in dynamic environments and its ability to learn new symbols by setting up this curriculum. We trained the agent sequentially on three levels. For GT, since it has already defined all prior knowledge needed in the environment, we directly used GT to train the final level to evaluate its performance. In Taxi, instead of curriculum learning, we trained the agent to learn abstraction directly from complete tasks.

Metrics. We recorded the average reward per episode for each batch. Mean results are shown as learning curves, smoothed with a 10-window moving average applied to the batch average rewards. Shaded areas indicate the 95% confidence interval of these average rewards.

5.2 Experimental Results

The experimental results for MiniGrid are shown in Figure 5. It can be observed that the performance of pure RL and RUDDER is not satisfactory, as they fail to converge directly in the final levels. For RUDDER, due to the high input dimensionality of the environment, it negatively impacts the training of the LSTM, resulting in poor performance in such a complex environment. GT starts training directly from the most challenging final level, and with a clear symbolic structure, it converges quickly and stably.

For A2RL, when the difficulty level begins/changes, the prior ASM initially struggles, reflected in very low rewards. A2RL then discovers new states and updates the ASM with new symbols, causing its performance to rapidly improve. It converges stably across maps of varying difficulty, demonstrating its ability to discover new states and effectively adapt to new environments. Additionally, the abstracted symbolic structures allow A2RL to outperform vanilla RL, which answers the questions 1 and 2 at the beginning of this section.

The experimental results for Taxi are shown in Figure 6. Vanilla RL matches GT-level performance but is significantly slower than A2RL, which rapidly abstracts passenger symbols and achieves similar results in fewer training steps. Addressing question 2, vanilla RL's single network struggles with prioritizing multiple objectives (e.g., passenger pick-up vs. destination) due to its limited capacity. In contrast, A2RL decomposes complex tasks into sub-tasks, allowing its individual atomic policies to master simpler objectives, thereby improving performance on more complex tasks.

5.3 Out-of-distribution Generalization

To address question 3, we conducted a series of ablation experiments: We replicated the setup in Section 5.2 but trained agents on procedurally generated random maps throughout the entire curriculum, and subsequently evaluated and fine-tuned on 50 unseen maps of comparable difficulty after training. We generated two distinct sets of Minigrid test maps: Minigrid-small and Minigrid-large.

Figure 7 shows that despite a slight decline in reward convergence speed and policy performance when training on test maps, the agent still achieves high performance. The generalization success rate on these test maps is below 10% (Generalization remains a significant challenge in DRL), yet learned atomic policies can complete some sub-tasks on many maps, as shown in Figure 7 (b). Furthermore, Figure 7 (c) and (d) illustrate that fine-tuning on test maps enables the agent to converge rapidly, typically within 50,000 steps, demonstrating strong generalization capabilities. These findings directly address question 3: A2RL enhances RL generalization by leveraging learned task structures and symbolic representations to generalize to similarly structured tasks. In comparison, vanilla DRL methods failed to converge on unseen maps.

6 Related Work

State Abstraction in RL. Early methods relied on expert-defined mappings to discrete state spaces [Dietterich, 2000; Andre and Russell, 2002], yet these struggle in continuous environments. End-to-end deep RL avoids explicit abstraction at the cost of reduced interpretability and limited generalization. To address this, Quentin et al. proposed NeSy to extract symbolic logic from neural strategies [Delfosse *et al.*, 2023], and Jiang et al. introduced NLRL, which leverages first-order logic to enhance both interpretability and task transferability [Jiang and Luo, 2019]. Other approaches learn abstractions from data [Furelos-Blanco *et al.*, 2023;

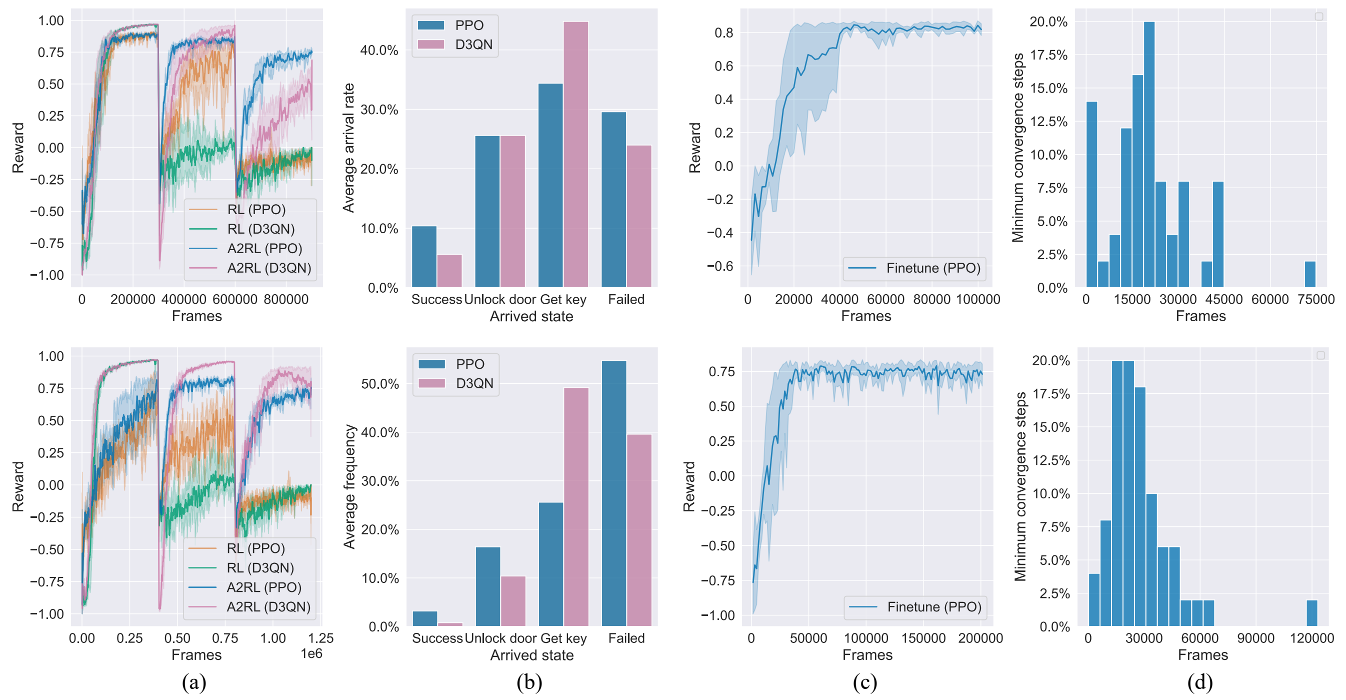


Figure 7: The two rows represent `Minigrid-small` and `Minigrid-large` with random maps. (a) The reward curves of A2RL are trained directly on 5 types of random maps. (b) The results of testing on 50 unseen maps after training on random maps, showing the proportion of sub-tasks completed on the 50 unfamiliar maps. For each of the 5 trained models, three tests were conducted on each map, and the best result among them was recorded. (c) The reward curve of A2RL during fine-tuning for generalization on the 50 unseen maps. (d) The distribution of the number of steps required for the reward curve to first achieve convergence during fine-tuning (convergence is defined as achieving success in more than 50 out of the most recent 100 episodes).

Parać *et al.*, 2024], but all depend on predefined symbols or labeled data, making automatic abstraction difficult. In contrast, A2RL can perform symbolic abstraction within the environment without predefined symbolic knowledge.

Knowledge Discovery in RL. Agents should infer "causal rule" knowledge from interacting with the environment. Some approaches use graph models or attention mechanisms to infer causal relations [peng *et al.*, 2022; Zhu *et al.*, 2020]; Ha et al. proposed world models for virtual trial-and-error [Ha and Schmidhuber, 2018], but struggle to acquire reliable rule-based knowledge and suffer from poor generalization. Existing methods fail to autonomously form symbols or discover causality in non-symbolic settings, and their probabilistic models lack logical reliability and expressive power. A2RL can autonomously discover new symbols within the environment and establish connections with existing symbols.

RL with Formal Knowledge. There are RL systems that use symbolic rule knowledge to decompose tasks through reward decomposition [Pateria *et al.*, 2021]. Recent efforts employ natural language instructions or Transformer architectures to guide HRL and capture long-range dependencies [Jiang *et al.*, 2019; K *et al.*, 2023; Chen *et al.*, 2021]. Alternative approaches combine classical planning with HRL to reduce its sample complexity [Yang *et al.*, 2018; Liu *et al.*, 2024]. Croonenborghs et al. investigated transfer learning within Relational Reinforcement Learning, which leverages existing knowledge to accelerate learning in target tasks; however, it is limited to tasks characterized by symbolic feature spaces [Croonenborghs *et al.*, 2007]. Most current methods encode knowledge as rewards without rule-based frameworks, and cannot dynamically acquire or revise symbolic knowledge. In contrast, A2RL can update symbolic structures through exploration within the environment.

7 Conclusion

In this paper, we present Abductive Abstract Reinforcement Learning (A2RL), a novel framework that integrates the strengths of symbolic abstraction with RL. A2RL enables agents to decompose complex tasks into manageable subtasks and perform abstractions directly from sub-symbolic environments without relying on predefined symbols. It is important to note that this work provides only a preliminary implementation of A2RL. Future research will focus on enhancing the abstraction mechanisms and expanding the application of A2RL to a wider range of real-world scenarios, such as training large language model (LLM)-based reasoners with a formal step-by-step reasoning module. Through the continued development of A2RL, we aim to further unify symbolic reasoning with learning, advancing the creation of more reliable and efficient AI systems capable of addressing complex and unstructured challenges.

Acknowledgements

The authors thank the anonymous reviewers of IJCAI 2025 for their valuable feedback. This research was supported by NSFC (62206124), Jiangsu Science Foundation Leading-edge Technology Program (BK20232003) and Major Program (JD) of Hubei Province (2023BAA024). This work is also funded by Nanjing University-China Mobile Communications Group Co., Ltd. Joint Institute.

Contribution Statement

*Wang-Zhou Dai is the corresponding author.
†Zilong Wang and Jiongda Wang contributed equally to this work and are considered co-first authors.

References

[Andre and Russell, 2002] David Andre and Stuart Russell. State abstraction for programmable reinforcement learning agents. In *Eighteenth National Conference on Artificial Intelligence*, page 119–125, Alberta, Canada, 2002. American Association for Artificial Intelligence.

[Arjona-Medina *et al.*, 2019] Jose A. Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. Rudder: Return decomposition for delayed rewards. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[Bai *et al.*, 2016] Aijun Bai, Siddharth Srivastava, and Stuart Russell. Markovian state and action abstractions for mdps via hierarchical MCTS. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 3029–3039, New York, NY, 2016. IJCAI/AAAI Press.

[Chen *et al.*, 2021] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. In *Advances in Neural Information Processing Systems*, volume 34, pages 15084–15097, 2021.

[Chevalier-Boisvert *et al.*, 2023] Maxime Chevalier-Boisvert, Bolun Dai, Mark Towers, Rodrigo de Lazcano, Lucas Willems, Salem Lahlou, Suman Pal, Pablo S. Castro, and Jordan Terry. Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. In *Advances in Neural Information Processing Systems*, volume 36, New Orleans, LA, 2023.

[Croonenborghs *et al.*, 2007] Tom Croonenborghs, Kurt Driessens, and Maurice Bruynooghe. Learning relational options for inductive transfer in relational reinforcement learning. In *Proceedings of the 17th International Conference on Inductive Logic Programming*, page 88–97, 2007.

[Dai and Muggleton, 2021] Wang-Zhou Dai and Stephen H. Muggleton. Abductive knowledge induction from raw data. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 1845–1851, Montreal, Canada, 2021. ijcai.org.

[Dai *et al.*, 2019] Wang-Zhou Dai, Qiu-Ling Xu, Yang Yu, and Zhi-Hua Zhou. Bridging machine learning and logical reasoning by abductive learning. In *Advances in Neural Information Processing Systems 32*, pages 2811–2822. Curran Associates, Inc., 2019.

[Delfosse *et al.*, 2023] Quentin Delfosse, Hikaru Shindo, Devendra Dhami, and Kristian Kersting. Interpretable and explainable logical policies via neurally guided symbolic abstraction. In *Advances in Neural Information Processing Systems*, volume 36, pages 50838–50858, 2023.

[Dietterich, 2000] Thomas G. Dietterich. Hierarchical reinforcement learning with the MAXQ value function decomposition. *Journal of artificial intelligence research*, 13(1):227–303, 2000.

[Furelos-Blanco *et al.*, 2023] Daniel Furelos-Blanco, Mark Law, Anders Jonsson, Krysia Broda, and Alessandra Russo. Hierarchies of reward machines. In *Proceedings of the Fortieth International Conference on Machine Learning*, ICML'23, Honolulu, Hawaii, 2023. JMLR.org.

[Gold and Shadlen, 2007] Joshua I. Gold and Michael N. Shadlen. The neural basis of decision making. *Annual review of neuroscience*, 30:535–74, 2007.

[Ha and Schmidhuber, 2018] David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[Hasselt *et al.*, 2016] Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 2094–2100, Phoenix, Arizona, 2016. AAAI Press.

[Jiang and Luo, 2019] Zhengyao Jiang and Shan Luo. Neural logic reinforcement learning. In *Proceedings of the Thirty-Six International Conference on Machine Learning*, volume 97, pages 3110–3119, 2019.

[Jiang *et al.*, 2019] YiDing Jiang, Shixiang (Shane) Gu, Kevin P Murphy, and Chelsea Finn. Language as an abstraction for hierarchical deep reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[K *et al.*, 2023] Namasivayam K, Himanshu Singh, Vishal Bindal, Arnav Tuli, Vishwajeet Agrawal, Rahul Jain, Parag Singla, and Rohan Paul. Learning neuro-symbolic programs for language guided robot manipulation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7973–7980, 2023.

[Kahneman, 2011] Daniel Kahneman. *Thinking, fast and slow*. Farrar, Straus and Giroux, New York, 2011.

[Kersting *et al.*, 2004] Kristian Kersting, Martijn Van Otterlo, and Luc De Raedt. Bellman goes relational. In *Proceedings of the Twenty-First International Conference on Machine Learning*, page 59, 2004.

[Korf, 1987] Richard E. Korf. Planning as search: a quantitative approach. *Artificial Intelligence*, 33(1):65–68, 1987.

[Ladosz *et al.*, 2022] Pawel Ladosz, Lilian Weng, Minwoo Kim, and Hyondong Oh. Exploration in deep reinforcement learning: A survey. *Information Fusion*, 85:1–22, 2022.

[Levy *et al.*, 2019] Andrew Levy, George Konidaris, Robert Platt, and Kate Saenko. Learning multi-level hierarchies with hindsight. In *The Seventh International Conference on Learning Representations*, 2019.

[Liu *et al.*, 2024] Jung-Chun Liu, Chi-Hsien Chang, Shao-Hua Sun, and Tian-Li Yu. Integrating planning and deep reinforcement learning via automatic induction of task substructures. In *The Twelfth International Conference on Learning Representations*, 2024.

[Mnih *et al.*, 2015] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[Parać *et al.*, 2024] Roko Parać, Lorenzo Nodari, Leo Ardon, Daniel Furelos-Blanco, Federico Cerutti, and Alessandra Russo. Learning robust reward machines from noisy labels. In *Proceedings of the Twenty-first International Conference on Principles of Knowledge Representation and Reasoning*, KR '24, Hanoi, Vietnam, 2024.

[Pateria *et al.*, 2021] Shubham Pateria, Budhitama Subagdja, Ah-hwee Tan, and Chai Quek. Hierarchical reinforcement learning: A comprehensive survey. *ACM Computing Surveys(CSUR)*, 54(5), 2021.

[peng *et al.*, 2022] shaohui peng, Xing Hu, Rui Zhang, Ke Tang, Jiaming Guo, Qi Yi, Ruizhi Chen, xishan zhang, Zidong Du, Ling Li, Qi Guo, and Yunji Chen. Causality-driven hierarchical structure discovery for reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 35, pages 20064–20076, 2022.

[Ratcliff and McKoon, 2008] Roger Ratcliff and Gail McKoon. The diffusion decision model: Theory and data for two-choice decision tasks. *Neural Computation*, 20:873–922, 2008.

[Russell, 2015] Stuart Russell. Unifying logic and probability. *Communications of the ACM*, 58(7):88–97, 2015.

[Schulman *et al.*, 2017] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

[Sutton and Barto, 2018] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, 2018.

[Towers *et al.*, 2024] Mark Towers, Ariel Kwiatkowski, Jordan Terry, John U Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun KG, et al. Gymnasium: A standard interface for reinforcement learning environments. *arXiv preprint arXiv:2407.17032*, 2024.

[Unruh and Rosenbloom, 1989] Amy Unruh and Paul S. Rosenbloom. Abstraction in problem solving and learning. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence - Volume 1*, pages 681–687, CA, USA, 1989. Morgan Kaufmann.

[Wang *et al.*, 2016] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando De Freitas. Dueling network architectures for deep reinforcement learning. In *Proceedings of the Thirty-Third International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 1995–2003, New York, NY, 2016. JMLR.org.

[Yang *et al.*, 2018] Fangkai Yang, Daoming Lyu, Bo Liu, and Steven Gustafson. Peorl: Integrating symbolic planning and hierarchical reinforcement learning for robust decision-making. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4860–4866, 2018.

[Zhou and Huang, 2021] Zhi-Hua Zhou and Yu-Xuan Huang. Abductive learning. In Pascal Hitzler and Md. Kamruzzaman Sarker, editors, *Neuro-Symbolic Artificial Intelligence: The State of the Art*, volume 342 of *Frontiers in Artificial Intelligence and Applications*, pages 353–369. IOS Press, 2021.

[Zhou, 2019] Zhi-Hua Zhou. Abductive learning: towards bridging machine learning and logical reasoning. *Science China Information Sciences*, 62(7), 2019.

[Zhu *et al.*, 2020] Shengyu Zhu, Ignavier Ng, and Zhitang Chen. Causal discovery with reinforcement learning. In *The Eighth International Conference on Learning Representations*, 2020.