HYPRL: Reinforcement Learning of Control Policies for Hyperproperties

Tzu-Han Hsu* **Arshia Rafieioskouei*** **Borzoo Bonakdarpour**
Michigan State University
{tzuhan, rafieios, borzoo}@msu.edu

Abstract

Reward shaping in multi-agent reinforcement learning (MARL) for complex tasks remains a significant challenge. Existing approaches often fail to find optimal solutions or cannot efficiently handle such tasks. We propose HYPRL, a specification-guided reinforcement learning framework that learns control policies w.r.t. *hyperproperties* expressed in HyperLTL. Hyperproperties constitute a powerful formalism for specifying objectives and constraints over sets of execution traces across agents. To learn policies that maximize the satisfaction of a HyperLTL formula φ, we apply Skolemization to manage quantifier alternations and define quantitative robustness functions to shape rewards over execution traces of a Markov decision process with unknown transitions. A suitable RL algorithm is then used to learn policies that collectively maximize the expected reward and, consequently, increase the probability of satisfying φ. We evaluate HYPRL on a diverse set of benchmarks, including safety-aware planning, Deep Sea Treasure, and the Post Correspondence Problem. We also compare with specification-driven baselines to demonstrate the effectiveness and efficiency of HYPRL.

1 Introduction

Designing reward functions that accurately capture desirable behaviors in multi-agent reinforcement learning (MARL) remains a notorious stumbling block. Consider a wildfire scenario in a 3×3 grid-world with cells labeled $\{a, b, \ldots, i\}$ (see Figure 1). Locations $\{i, f, c\}$ are on fire, and two victims are located at $\{f, g\}$. Now, two autonomous agents are deployed from $\{a\}$ with two objectives; $\boldsymbol{O_1}$: the firefighter agent (FF) must extinguish all fire zones, and $\boldsymbol{O_2}$: the medical agent (Med) aims to rescue all victims. The agents also have to satisfy two constraints; $\boldsymbol{C_1}$: they must always remain within a 2-cell communication range, and $\boldsymbol{C_2}$: Med cannot pass any fire zone before FF extinguished the fire in that zone. The goal is to learn optimal policies for FF and Med that maximize the probability of satisfying all above requirements, as shown by the agent paths in Figure 1a.

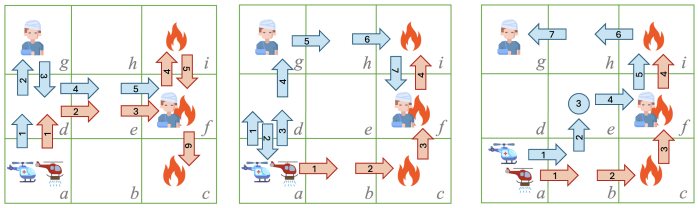
Now, suppose we use an existing RL approach by assigning the following rewards; extinguish fire: $+50$, rescue a victim: $+10$, agent out of range: -100, and Med in fire zone: -100. These approaches would guide FF to complete O_1 optimally by path $a \xrightarrow{\mathrm{R}} b \xrightarrow{\mathrm{R}} c \xrightarrow{\mathrm{U}} f \xrightarrow{\mathrm{U}} i$, but due to C_1, it forces Med to delay the rescue of the victim in $\{g\}$ with redundant moves: $a \xrightarrow{\mathrm{U}} d \xrightarrow{\mathrm{D}} a \xrightarrow{\mathrm{U}} d \xrightarrow{\mathrm{U}} g \cdots$ (see Figure 1b). Furthermore, the victim in $\{f\}$ is trapped in fire, so satisfying O_2 depends on the progress of O_1, while respecting C_1 (safety) and C_2 (dependency). Such complex requirements make reward design in MARL particularly challenging, as policies must account for both individual objectives and relational constraints between agents with dependencies. In fact, existing RL approaches often fail to find optimal policies because agent paths are implicitly *universally* quantified, which is overly restrictive and limits the ability of RL to explore more optimal solutions. That is, the set of requirements is of the form $\forall\tau.\forall\tau'.\psi$, where τ and τ' are agents' paths and ψ specifies constrains and objectives.

*Equal contribution.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

 https://doi.org/10.52202/085713-1743

◇ (O_1): FF extinguishes all fires.
◇ (O_2): Med rescues all victims.
◇ (C_1): FF and Med always stay within 2-cells of each other.
◇ (C_2): Med must not passing any fire zone at all time.



(a) Optimal policies. **(b)** Med paces in $a - d$. **(c)** Med pauses in e.

Figure 1: A wildfire scenario with two objectives (O_1, O_2) and two relational constraints (C_1, C_2).

Temporal logics [37] have shown to be both expressive and effective to address objectives and constraints in single-agent settings [32, 24, 2, 10, 14, 21, 22, 28, 45, 47]. However, these methods cannot handle our motivating example as it involves multiple agents. Recent works have been extended to MARL [33, 31, 20, 16], but these approaches either fail to capture relational dependencies between agents or can handle only a subset of temporal properties, such as *co-safty* [29, 41]. Moreover, specifications in temporal logics that formalize the behavior of individual traces (e.g., LTL) are again inherently universally quantified and, hence, too restrictive.

In this paper, we propose a specification-guided RL framework that leverages the powerful framework of *hyperproperties* [12] and, in particular, the temporal logic HyperLTL [13], as its formal foundation. Hyperproperties characterize requirements over sets of execution traces, allowing the specification of relational and dependent behaviors that involve multiple agents or joint system executions, rather than individual behaviors of agents. This expressiveness is particularly well-suited for capturing relational, quantified, and multi-agent objectives. Building on this foundation, we propose **Hyp**erproperties for **R**einforcement **L**earning (HYPRL), a framework that learns a collection of control policies which maximizes the probability of satisfying a hyperproperty expressed as a HyperLTL formula. In our motivating example, HyperLTL allows a specification of the form $\forall\tau.\exists\tau'.\psi$, which enables HYPRL to search for more optimal policies and identify the one in Figure 1a. The main challenge here is to deal with HyperLTL formulas with quantifier alternation, which has not been studied prior to HYPRL. Our work represents a novel step towards the automated synthesis of reward mechanisms that capture complex objectives and constraints in multi-agent setting.

Contributions:

1. We formulate the control policies synthesis problem for a multi-agent system as a learning problem, where the goal is to maximize the probability of satisfying a HyperLTL formula that expresses a set of objectives and constraints (Section 4).
2. We address the challenge of reasoning about quantifier alternation in a HyperLTL formula due to expressing dependencies among agents by Skolemization (Section 5.1).
3. We introduce quantitative semantics for HyperLTL that compute robustness values as rewards, providing a solution of reward shaping for optimizing hyperproperty satisfaction (Section 5.2). Finally, we use off-the-shelf RL algorithms to learn an optimal collection of policies (Section 5.3).
4. We evaluate HYPRL on a diverse set of learning tasks, including safety-aware multi-objective planning, the Post Correspondence Problem, the famous Deep Sea Treasure benchmark, and the wildfire scenario. Our results show that HYPRL is more efficient and effective in handling complex requirements compared to selected baselines (Section 6).

2 Related Work

Logic-based Single-agent RL. DIRL [24] is a framework that synthesizes an optimal policy which the specification is expressed in the language SPECTRL [23]. However, SPECTRL does not capture the full expressiveness of temporal logics as it only supports a fragment of LTL. In [32], the authors introduce *truncated* LTL for quantitative reasoning over LTL formulas, but it is limited to single-agent task specifications (similar reasonings are also presented in [44, 2, 10]). In [28], LTL formulas are encoded by replacing operators and predicates with recurrent neural networks connected according to the parse tree, forming a structure that captures the formula's semantics. In [14, 21, 45, 47, 30], the authors compose an MDP with a Limit Deterministic Büchi Automaton (LDBA), which represents the LTL specification, and solve the compositional model as a control problem.

Logic-based Multi-Agent Reinforcement Learning. In [25], the authors extend the SPECTRL language [23] to non-cooperative multi-agent systems and train joint policies that form a Nash Equilibrium, which requires relational reasoning. However, their algorithm follows an "enumerate-and-verify" approach by exhaustively searching RL policies to identify one that has higher probability to form such equilibrium, which leads to substantial learning overhead as the number of agents increases. Moreover, the extended language in [25] cannot handle arbitrary HyperLTL formulas. In [16], the authors propose a reward shaping method by composing an MDP with a single centralized LDBA derived from a global LTL specification. However, this approach cannot capture inter-agent dependencies, and the product construction of LDBA may suffer from state-space explosion. More recent efforts in [33, 31, 20] extend quantitative reasoning to formulas with LTL operators in a multi-agent setting. However, in contrast to HYPRL, they only cover *co-safety* specifications and do not support the full logic.

Shield Synthesis. Shield synthesis for RL is a technique that asks an agent to propose an action in each learning step, and a *shield* (i.e., a safety guard) evaluates whether such action is safe [3, 26, 27]. In [34], the authors apply shield synthesis in a decentralized multi-agent setting, where the learning targets are specified by deterministic finite automata. However, the specifications are limited to universal (i.e., a $\forall^*.\psi$ formula) and cannot handle properties such as planning tasks that involve dependencies (which is a $\forall\exists$ hyperproperty). Furthermore, the authors in [17] proposed *factored* shielding, which can learn multiple policies by a factorization of joint state space (i.e., decomposing one shield into multiple sub-shields). However, the main contribution is the improvement on RL scalability, but the limitation on universal-only properties remains.

3 Preliminaries

Markov Decision Processes (MDP). We model our RL problem for hyperproperties using MDP $\mathcal{M} = \langle S, s^0, A, \mathbf{P}, \mathsf{AP}, L\rangle$, where S is a finite set of *states*, $s^0 \in S$ is the *initial state*, and A is a finite set of *actions*. The *transition probability function* $\mathbf{P} : S \times A \times S \to [0, 1]$ assigns a probability to each transition, ensuring that for any state s and action a, the sum of outgoing probabilities satisfies $\sum_{s' \in S} \mathbf{P}(s, a, s') = 1$. Additionally, AP denotes a finite set of *atomic propositions*, and $L : S \to 2^{\mathsf{AP}}$ is the *labeling* function. For example, Figure 2 represents the MDP of Figure 1, where each state can be represented by $\langle x, y\rangle$ with $x, y \in \{0, 1, 2\}$, labeled by $\{a, b, ..., i\}$, and actions $A = \{U, D, L, R\}$ (i.e., four moving directions). A *path* in $\mathcal{M}$ is a sequence $\zeta = s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} \cdots$, where $s_i \in S$ and $a_i \in A$ for all $i \geq 0$. A *sub-path* $\zeta_{[\ell:k]}$ is a segment $s_\ell \xrightarrow{a_\ell} \cdots \xrightarrow{a_{k-1}} s_k$, for $0 \leq \ell < k < |\zeta|$. We write $\mathcal{Z}^*$ and $\mathcal{Z}^\omega$ to denote the sets of all finite and infinite paths, respectively. A (deterministic) *policy* $\pi : \mathcal{Z}^* \to A$ maps each finite path to a (fixed) action $a \in A$. The *trace* t of a path ζ is the sequence of labels $\mathsf{Tr}(\zeta) = t(0)t(1)t(2)\cdots$, where $t(i) = L(s_i)$ for all $i \geq 0$. Slightly abusing notation, we use $\mathsf{Traces}(\mathcal{Z}^*)$ and $\mathsf{Traces}(\mathcal{Z}^\omega)$ to denote the sets of all finite and infinite traces, respectively.

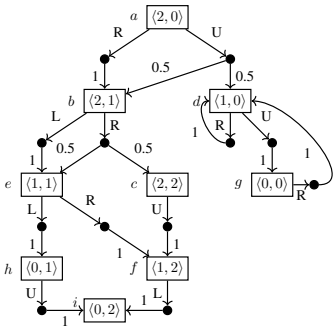


Figure 2: The MDP of The grid-world in Figure 1.

Finite Semantics for HyperLTL. The syntax of HyperLTL [13] is given by the following grammar:

$$\varphi ::= \exists \tau.\varphi \mid \forall \tau.\varphi \mid \psi \qquad \psi ::= p_\tau \mid \neg\psi \mid \psi \vee \psi \mid \bigcirc\psi \mid \psi\, \mathcal{U}\, \psi,$$

where $p \in \mathsf{AP}$ is an atomic proposition, τ is a *trace variable*, and the second rule (for unquantified formulas) produces LTL formulas. The Boolean connectives $\neg$ and $\vee$ have their standard meanings, while $\bigcirc$ and $\mathcal{U}$ denote the "next" and "until" temporal operators, respectively. Other Boolean and temporal operators are derived as syntactic sugar: $\mathsf{true} \triangleq p_\tau \vee \neg p_\tau$, $\mathsf{false} \triangleq \neg\mathsf{true}$, $\psi_1 \to \psi_2 \triangleq \neg\psi_1 \vee \psi_2$, $\Diamond\psi \triangleq \mathsf{true}\ \mathcal{U}\ \psi$, and $\Box\psi \triangleq \neg\Diamond\neg\psi$, where '$\Diamond$' and '$\Box$' are the *eventually* and *always* temporal operators. In a quantified formula, $\exists\tau$ means "along some trace τ", and $\forall\tau$ means "along all traces τ". We write $Vars(\varphi)$ for the set of trace variables in a formula φ. A formula is *closed* if all $\tau \in Vars(\varphi)$ are quantified, with no variable quantified twice. Since RL algorithms operate on finite samples, we adopt the *finite semantics* of HyperLTL [11] over finite trace assignments, where a partial mapping $\Pi : Vars(\varphi) \rightharpoonup (2^{\mathsf{AP}})^*$ assigns each $\tau \in Vars(\varphi)$ to a finite trace. Given Π, a trace variable τ, and a finite trace $t \in (2^{\mathsf{AP}})^*$, we write $\Pi[\tau \to t]$ for the assignment identical to Π, except that τ is mapped to t. We denote by $\Pi_\emptyset$ the empty trace assignment. For $t \in \Pi$, we refer to traces in the image of Π. Throughout this paper, we abbreviate tuples $\langle x_1, \ldots, x_n\rangle$ as $\langle x_i\rangle_{i\in 1,\ldots,n}$. An *interpretation* of

https://doi.org/10.52202/085713-1743

a HyperLTL formula $\varphi = \mathbb{Q}_1\tau_1 \ldots \mathbb{Q}_n\tau_n.\ \psi$, denoted by $\mathcal{T} = \langle T_{\tau_i} \rangle_{i \in \{1,\ldots,|\,Vars(\varphi)|\}}$, is a tuple of sets of traces, where we have one set T_{τ_i} per trace variable τ_i, denoting the set of traces that can be assigned to τ_i. Let $\mathcal{D}_{\pi_i}$ be the distribution over a set of paths induced by a policy π_i, and we write $\mathcal{Z}_{\tau_i} \sim \mathcal{D}_{\pi_i}$ to denote a set of paths $\mathcal{Z}_{\tau_i}$ sampled from $\mathcal{D}_{\pi_i}$, such that π_i is the policy associated with the trace variable τ_i, for each $i \in \{1, \ldots, |\,Vars(\varphi)|\}$ (each trace variable ranges over the possible behaviors of an agent). We also define a *family* of sampled sets $\mathcal{S} = \langle \mathcal{Z}_{\tau_i} \sim \mathcal{D}_{\pi_i} \rangle_{i \in \{1,\ldots,|\,Vars(\varphi)|\}}$. That is, for each τ_i in $Vars(\varphi)$, $T_{\tau_i} = \mathsf{Traces}(\mathcal{Z}_{\tau_i} \sim \mathcal{D}_{\pi_i})$ is the set of traces that τ_i can range over, which comes from the sampled paths from the associated policy π_i. Abusing notation, we write $\mathcal{T} = \mathsf{Traces}(\mathcal{S})$ as the tuple of sets of sampled traces. The satisfaction relation $\models$ maps a formula φ to a model $(\mathcal{T}, \Pi, i)$, where $i \in \mathbb{Z}_{\geq 0}$ indicates the current evaluation position. Formally:

$$
\begin{array}{llll}
(\mathcal{T}, \Pi, 0) & \models \exists\tau.\ \psi & \text{iff} & \text{there is a } t \in T_\tau, \text{ such that } (\mathcal{T}, \Pi[\tau \to t], 0) \models \psi \\
(\mathcal{T}, \Pi, 0) & \models \forall\tau.\ \psi & \text{iff} & \text{for all } t \in T_\tau, \text{ such that } (\mathcal{T}, \Pi[\tau \to t], 0) \models \psi \\
(\mathcal{T}, \Pi, i) & \models p_\tau & \text{iff} & p \in \Pi(\tau)(i) \\
(\mathcal{T}, \Pi, i) & \models \neg\psi & \text{iff} & (\mathcal{T}, \Pi, i) \not\models \psi \\
(\mathcal{T}, \Pi, i) & \models \psi_1 \vee \psi_2 & \text{iff} & (\mathcal{T}, \Pi, i) \models \psi_1 \text{ or } (\mathcal{T}, \Pi, i) \models \psi_2 \\
(\mathcal{T}, \Pi, i) & \models \bigcirc \psi & \text{iff} & (\mathcal{T}, \Pi, i+1) \models \psi \text{ and for all } t \in \Pi.|t| \geq i+1 \\
(\mathcal{T}, \Pi, i) & \models \psi_1\, \mathcal{U}\, \psi_2 & \text{iff} & \text{there exists } j \geq i \text{ with } j < \min_{t \in \Pi} |t|, \text{such that } (\mathcal{T}, \Pi, j) \models \psi_2 \\
 & & & \text{and for all } k \in [i, j), (\mathcal{T}, \Pi, k) \models \psi_1.
\end{array}
$$

We say that an interpretation $\mathcal{T}$ satisfies a HyperLTL formula φ, written as $\mathcal{T} \models \varphi$, if $(\mathcal{T}, \Pi_\emptyset, 0) \models \varphi$. Likewise, a family of samples $\mathcal{S}$ (induced by each π_{τ_i} associated with each $\tau_i \in Vars(\varphi)$), satisfies a sentence φ if $\langle \mathsf{Traces}(\mathcal{Z}_{\tau_i} \sim \mathcal{D}_{\pi_i}) \rangle_{i \in \{1,\ldots,|\,Vars(\varphi)|\}} \models \varphi$.

Example. The following HyperLTL formula captures the objectives and constraints of our motivating example described in Figure 1, where τ_1 is the path for FF and τ_2 is the path for Med:

$$
\textit{Specification}: \ \varphi_{\mathsf{Rescue}} \triangleq \forall\tau_1.\exists\tau_2.(\psi_{\mathsf{fire}} \wedge \psi_{\mathsf{save}} \wedge \psi_{\mathsf{dist}} \wedge \psi_{\mathsf{safe}})
$$

$$
O_1: \ \psi_{\mathsf{fire}} \triangleq \Diamond(i_{\tau_1}) \wedge \Diamond(f_{\tau_1}) \wedge \Diamond(c_{\tau_1}) \qquad C_1: \ \psi_{\mathsf{dist}} \triangleq \Box(|\mathsf{Location}_{\tau_1} - \mathsf{Location}_{\tau_2}| < 3)
$$

$$
O_2: \ \psi_{\mathsf{save}} \triangleq \Diamond(g_{\tau_2}) \wedge \Diamond(f_{\tau_2}) \qquad C_2: \ \psi_{\mathsf{safe}} \triangleq (\neg i_{\tau_2}\, \mathcal{U}\, i_{\tau_1}) \wedge (\neg f_{\tau_2}\, \mathcal{U}\, f_{\tau_1}) \wedge (\neg c_{\tau_2}\, \mathcal{U}\, c_{\tau_1})
$$

For instance, the dependency constraint C_2 for "Med cannot enter any fire zone until FF extinguished the fire in that zone" is expressed using the conjunction of temporal *until* operators. Notice that $\varphi_{\mathsf{Rescue}}$ features $\forall\exists$ quantifier alternation, which increases the complexity of reasoning about hyperproperties [8]. We emphasize that most RL approaches assume purely universal forms (i.e., $\forall^*$), which cannot capture agent dependencies and often yield sub-optimal solutions. For instance, if FF ignores that Med must wait until the fire is extinguished to rescue a victim, FF may follow its own optimal path that causes unnecessary delay for Med (see Figure 1c).

4 Problem Statement

Let us use $\star$ to denote optimality (e.g., $\pi^\star$ denotes an optimal policy). The following optimization problem formulates policy synthesis as a learning problem.

Given an MDP $\mathcal{M}$ with unknown transitions and a HyperLTL formula φ of the form $\mathbb{Q}_1\tau_1 \ldots \mathbb{Q}_n\tau_n.\ \psi$, our goal is to identify a tuple of n policies $\langle \pi_1^\star, \ldots, \pi_n^\star \rangle$, such that:

$$
\langle \pi_i^\star \rangle_{i \in \{1,\ldots,n\}} \in \left[\arg\max_{\langle \pi_i \rangle} \mathbb{P}\Big[\langle \mathsf{Traces}(\mathcal{Z}_{\tau_i} \sim \mathcal{D}_{\pi_i}) \rangle \models \varphi \Big] \right]_{i \in \{1,\ldots,n\}}
$$

where $\mathcal{D}_{\pi_1}, \ldots, \mathcal{D}_{\pi_n}$ are the distributions over set of paths generated by policy spaces $\pi_1, \ldots, \pi_n$. That is, $\langle \pi_1^\star, \ldots, \pi_n^\star \rangle$ maximizes the probability $\mathbb{P}$ such that the generated tuple of sets of traces $\langle \mathsf{Traces}(\mathcal{Z}_{\tau_1} \sim \mathcal{D}_{\pi_1}), \ldots, \mathsf{Traces}(\mathcal{Z}_{\tau_n} \sim \mathcal{D}_{\pi_n}) \rangle$ from $\mathcal{M}$ satisfies φ.

Example. Consider the MDP in Figure 2 and the following HyperLTL formula:

$$
\varphi_{\mathsf{exp}} \triangleq \forall\tau_1.\exists\tau_2.\Big(\Diamond i_{\tau_1} \wedge \Box\, dist(\langle x, y \rangle_{\tau_1}, \langle x, y \rangle_{\tau_2}) < 3 \Big)
$$

Suppose agent FF with π_1 draws samples $\mathcal{Z}_{\tau_1} = \{\zeta^1_{\mathsf{FF}}, \zeta^2_{\mathsf{FF}}\}$ from the MDP:

$$
\zeta^1_{\mathsf{FF}}: \underbrace{\langle 2,0 \rangle}_{a} \xrightarrow{\mathrm{R}} \underbrace{\langle 2,1 \rangle}_{b} \xrightarrow{\mathrm{R}} \underbrace{\langle 2,2 \rangle}_{c} \xrightarrow{\mathrm{U}} \underbrace{\langle 1,2 \rangle}_{f} \xrightarrow{\mathrm{L}} \underbrace{\langle 0,2 \rangle}_{i} \qquad \zeta^2_{\mathsf{FF}}: \underbrace{\langle 2,0 \rangle}_{a} \xrightarrow{\mathrm{R}} \underbrace{\langle 2,1 \rangle}_{b} \xrightarrow{\mathrm{R}} \underbrace{\langle 1,1 \rangle}_{e} \xrightarrow{\mathrm{L}} \underbrace{\langle 0,1 \rangle}_{h} \xrightarrow{\mathrm{U}} \underbrace{\langle 0,2 \rangle}_{i}
$$

Agent Med with policy π_2 draws $\mathcal{Z}_{\tau_2} = \{\zeta^1_{\mathsf{Med}}, \zeta^2_{\mathsf{Med}}\}$:

$$\zeta^1_{\mathsf{Med}} : \underbrace{\langle 2,0\rangle}_{a} \xrightarrow{\mathrm{U}} \underbrace{\langle 1,0\rangle}_{d} \xrightarrow{\mathrm{U}} \underbrace{\langle 0,0\rangle}_{g} \xrightarrow{\mathrm{R}} \underbrace{\langle 1,0\rangle}_{d} \xrightarrow{\mathrm{R}} \underbrace{\langle 1,0\rangle}_{d} \qquad \zeta^2_{\mathsf{Med}} : \underbrace{\langle 2,0\rangle}_{a} \xrightarrow{\mathrm{U}} \underbrace{\langle 2,1\rangle}_{b} \xrightarrow{\mathrm{L}} \underbrace{\langle 1,1\rangle}_{e} \xrightarrow{\mathrm{R}} \underbrace{\langle 1,2\rangle}_{f} \xrightarrow{\mathrm{L}} \underbrace{\langle 0,2\rangle}_{i}$$

Notice that, the number of samples can be more than two. We now calculate the probability of satisfying φ using $\mathcal{Z}_{\tau_1}$ and $\mathcal{Z}_{\tau_2}$ (obtained by π_1 and π_2) as follows:

$$\mathsf{Traces}(\langle\{\zeta^1_{\mathsf{FF}}\}, \mathcal{Z}_{\tau_2}\rangle) \models \varphi_{\mathsf{exp}} \quad \mathsf{Traces}(\langle\{\zeta^2_{\mathsf{FF}}\}, \mathcal{Z}_{\tau_2}\rangle) \models \varphi_{\mathsf{exp}},$$

where ζ^2_{Med} is the witness to τ_2 (existentially quantified) for both satisfaction relations. Hence, given $\mathcal{Z}_1$ and $\mathcal{Z}_2$, the probability of satisfying φ_{exp} is evaluated as:

$$\mathbb{P}_{\langle\pi_1,\pi_2\rangle}\Big[\mathsf{Traces}(\langle\mathcal{Z}_{\tau_1}, \mathcal{Z}_{\tau_2}\rangle) \models \varphi_{\mathsf{exp}}\Big] = 1$$

Now, if φ_{exp} had the form $\forall\forall$, the evaluation has to go over all combinations between $\mathcal{Z}_{\tau_1}$ and $\mathcal{Z}_{\tau_2}$:

$$\mathsf{Traces}(\langle\{\zeta^1_{\mathsf{FF}}\}, \{\zeta^1_{\mathsf{Med}}\}\rangle) \not\models \varphi_{\mathsf{exp}} \quad \mathsf{Traces}(\langle\{\zeta^1_{\mathsf{FF}}\}, \{\zeta^2_{\mathsf{Med}}\}\rangle) \models \varphi_{\mathsf{exp}}$$
$$\mathsf{Traces}(\langle\{\zeta^2_{\mathsf{FF}}\}, \{\zeta^1_{\mathsf{Med}}\}\rangle) \not\models \varphi_{\mathsf{exp}} \qquad \mathsf{Traces}(\langle\{\zeta^2_{\mathsf{FF}}\}, \{\zeta^2_{\mathsf{Med}}\}\rangle) \models \varphi_{\mathsf{exp}}$$

Thus, the satisfaction probability of $\forall\forall\psi$ would be 0.5. This example demonstrates that the probability of satisfying a HyperLTL formula crucially depends on its quantifier structure.

5 Algorithmic Details of HYPRL

To solve the problem formally stated in Section 3, our algorithm proceeds in the following three steps. We first Skolemize φ [40] to eliminate quantifier alternations and simplify the learning task. We then define quantitative semantics for HyperLTL, converting satisfaction checking into robustness value optimization. Finally, we train a neural network using these robustness signals to learn optimal policies that solve the original learning problem.

5.1 Step 1: HyperLTL Skolemization

Let a HyperLTL formula be of the form $\varphi = \mathbb{Q}_1\tau_1.\mathbb{Q}_2\tau_2.\ldots\mathbb{Q}_n.\tau_n.\ \psi(\tau_1,\tau_2,\ldots,\tau_n)$, where for $1 \le \ell \le n$, each $\mathbb{Q}_\ell \in \{\forall,\exists\}$ quantifies a trace variable τ_ℓ, and ψ is a quantifier-free LTL formula. We first Skolemize φ, producing $\mathbf{Skolem}(\varphi)$, to eliminate quantifier alternations. We define $\mathbb{Q}^\exists = \{i \mid \mathbb{Q}_i = \exists\}$ and $\mathbb{Q}^\forall = \{j \mid \mathbb{Q}_j = \forall\}$ as the sets of existential and universal quantifier indices, respectively. For each $i \in \mathbb{Q}^\exists$, we denote $\mathbb{Q}^\forall_i = \{j < i \mid \mathbb{Q}_j = \forall\}$ as the index set of all universal quantifiers preceding $\mathbb{Q}_i$. A *Skolem function* for each $i \in \mathbb{Q}^\exists$ is defined as $\mathbf{f}_i : \mathcal{T}^{|\mathbb{Q}^\forall_i|} \to \mathcal{T}$, and reduces to a constant function when $\mathbb{Q}^\forall_i = \emptyset$. A trace assignment Π is *consistent* with $\mathbf{f}_i$, if $\Pi(\tau_{i_j}) \in \mathcal{T}$ for all $j \in \mathbb{Q}^\forall_i$, and $\Pi(\tau_i) = \mathbf{f}_i\big(\Pi(\tau_{i_1}), \Pi(\tau_{i_2}), \ldots, \Pi(\tau_{i_{|\mathbb{Q}^\forall_i|}})\big)$ for all $i \in \mathbb{Q}^\exists$, where $\mathbb{Q}^\forall_i = \{i_1 < i_2 < \cdots < i_{|\mathbb{Q}^\forall_i|}\}$. If $(\mathcal{T}, \Pi, 0) \models \varphi$ for every trace assignment Π consistent with all $\mathbf{f}_i$, then each $\mathbf{f}_i$ is said to *witness* the satisfaction of φ [43]. For the inner LTL formula ψ (i.e., obtaining $\mathbf{Skolem}(\psi)$), we replace each proposition p_{τ_i} with $p_{\mathbf{f}_i}$ for all $p \in \mathsf{AP}$ and $i \in \mathbb{Q}^\exists$, thereby instantiating variables of the existentially quantified paths via their Skolem witnesses. In general, a Skolemized φ is of the following form:

$$\mathbf{Skolem}(\varphi) = \underbrace{\exists \mathbf{f}_i(\tau_{i_1}, \ldots, \tau_{i_{|\mathbb{Q}^\forall_i|}}).}_{\text{for each } i \in \mathbb{Q}^\exists} \quad \underbrace{\forall \tau_j.}_{\text{for each } j \in \mathbb{Q}^\forall} \quad \mathbf{Skolem}(\psi) \tag{1}$$

Based on this transformation, we re-write the problem statement from Section 3 as follows. We first define the *image* of $\mathbf{f}_i$:

$$\mathit{Img}(\mathbf{f}_i) \triangleq \{\mathbf{f}_i(t_{i_1}, \ldots, t_{i_{|\mathbb{Q}^\forall_i|}}) \mid t_{i_j} \in \mathsf{Traces}(\mathcal{Z}_{\tau_{i_j}} \sim \mathcal{D}_{\pi_{i_j}}),\ j \in \mathbb{Q}^\forall_i\}$$

That is, $\mathit{Img}(\mathbf{f}_i)$ is the set of mapped traces (which τ_{i_j} for each $i \in \mathbb{Q}^\exists$ ranges over) from all possible preceding $\forall$-quantified τ_{i_j}, where each t_{i_j} is from its own sampled trace set $\mathcal{Z}_{\tau_{i_j}}$. Now, let us use $\bowtie$ as a notation to ensure the collection of trace sets are ordered w.r.t. their path indices. Given two tuples of sets of traces $\mathcal{T}_1$ and $\mathcal{T}_2$, we define $\mathcal{T}_1 \bowtie \mathcal{T}_2 \triangleq \langle\mathsf{Traces}(\mathcal{Z}_{\tau_x})\rangle_{x\in\{1\cdots n\}}$, where each

https://doi.org/10.52202/085713-1743

$$
\begin{array}{lcl}
\rho\big(\mathsf{Tr}(\zeta_{[\ell:k]}), \psi\big) & = & \rho_{min} \text{ if } \mathsf{Tr}(\zeta_{[\ell:..]}) = \epsilon \text{ and } \rho\big(\mathsf{Tr}(\zeta_{[\ell:k]}), \psi\big) \text{ otherwise.} \\
\rho\big(\mathsf{Tr}(\zeta_{[\ell:k]}), \mathsf{true}\big) & = & \rho_{max} \\
\rho\big(\mathsf{Tr}(\zeta_{[\ell:k]}), f\big(L(s_\ell) < c\big)\big) & = & c - f\big(L(s_\ell)\big) \\
\rho\big(\mathsf{Tr}(\zeta_{[\ell:k]}), \neg\psi\big) & = & -\rho\big(\mathsf{Tr}(\zeta_{[\ell:k]}), \psi\big) \\
\rho\big(\mathsf{Tr}(\zeta_{[\ell:k]}), \bigcirc\psi\big) & = & \rho\big(\mathsf{Tr}(\zeta_{[\ell+1:k]}), \psi\big) \text{ if } (k > \ell). \\
\rho\big(\mathsf{Tr}(\zeta_{[\ell:k]}), \square\psi\big) & = & \min\limits_{i\in[\ell,k)} \rho\big(\mathsf{Tr}(\zeta_{[i:k]}), \psi\big) \\
\rho\big(\mathsf{Tr}(\zeta_{[\ell:k]}), \diamondsuit\psi\big) & = & \max\limits_{i\in[\ell,k)} \rho\big(\mathsf{Tr}(\zeta_{[i:k]})\psi\big) \\
\rho\big(\mathsf{Tr}(\zeta_{[\ell:k]}), \psi_1 \wedge \psi_2\big) & = & \mathsf{min}\big(\rho\big(\mathsf{Tr}(\zeta_{[\ell:k]}), \psi_1\big)\rho\big(\mathsf{Tr}(\zeta_{[\ell:k]}), \psi_2\big)\big) \\
\rho\big(\mathsf{Tr}(\zeta_{[\ell:k]}), \psi_1 \vee \psi_2\big) & = & \mathsf{max}\big(\rho\big(\mathsf{Tr}(\zeta_{[\ell:k]}), \psi_1\big)\rho\big(\mathsf{Tr}(\zeta_{[\ell:k]}), \psi_2\big)\big) \\
\rho\big(\mathsf{Tr}(\zeta_{[\ell:k]}), \psi_1 \,\mathcal{U}\, \psi_2\big) & = & \max\limits_{i\in[\ell,k)} \Big(\mathsf{min}\Big(\rho\big(\mathsf{Tr}(\zeta_{[i:k]}), \psi_2\big), \min\limits_{j\in[\ell,i)} \rho\big(\mathsf{Tr}(\zeta_{[j:i]}), \psi_1\big)\Big)\Big)
\end{array}
$$

Figure 3: Quantitative semantics for LTL.

$\mathsf{Traces}(\mathcal{Z}_{\tau_x})$ is either in $\mathcal{T}_1$ or $\mathcal{T}_2$ (and not both). Given an MDP $\mathcal{M}$ and a HyperLTL specification φ of the form $\mathbb{Q}_1\tau_1.\mathbb{Q}_2\tau_2.\ldots.\mathbb{Q}_n\tau_n.\ \psi$, our goal is to compute (1) a tuple of Skolem witnesses $\langle \mathbf{f}_i \rangle_{i\in\mathbb{Q}^\exists}$, and (2) a tuple of policies $\langle \pi_j^\star \rangle_{j\in\mathbb{Q}^\forall}$, such that:

$$
\langle \pi_j^\star \rangle_{j\in\mathbb{Q}^\forall} \in \Big[\arg\max_{\langle\pi_j\rangle} \mathbb{P}\Big[\langle Img(\mathbf{f}_i)\rangle \bowtie \langle \mathsf{Traces}(\mathcal{Z}_{\tau_j} \sim \mathcal{D}_{\pi_j})\rangle \models \mathbf{Skolem}(\varphi)\Big]\Big]_{i\in\mathbb{Q}^\exists, j\in\mathbb{Q}^\forall}
$$

That is, the tuple of policies $\langle \pi_j^\star \rangle$ maximizes the probability that the ordered collection of (1) the generated traces of all universal quantifiers $\langle \mathsf{Traces}(\mathcal{Z}_{\tau_j} \sim \mathcal{D}_{\pi_j})\rangle_{j\in\mathbb{Q}^\forall}$ and (2) the sets of mapped traces (i.e., the image) of each Skolem witness for all existential quantifiers $\langle Img(\mathbf{f}_i)\rangle_{i\in\mathbb{Q}^\exists}$ together satisfies $\mathbf{Skolem}(\varphi)$. Notice that in the updated problem statement, we compute policies only for universally quantified traces, while Skolem functions are learned for existentially quantified traces to witness the optimality.

5.2 Step 2: Policy Learning with Quantitative Semantics

To transform the satisfaction checking problem (i.e., determining $\models$) into an optimization task, we define quantitative semantics for HyperLTL, extended from [32]. In particular, we evaluate the Skolemized HyperLTL formula $\mathbf{Skolem}(\varphi)$ over tuples of sampled paths $\langle \zeta_1, \zeta_2, \ldots, \zeta_n \rangle$ from $\mathcal{M}$.

Robustness for a Single Trace. Let $\mathbb{R}$ be the set of real numbers and Ψ the set of all LTL formulas. We define a valuation function $f : 2^{\mathsf{AP}} \to \mathbb{R}$ that assigns a real value to a set of atomic propositions, provided as part of the input. Given a state $s \in S$ of an MDP $\mathcal{M}$, the quantitative semantics are defined over predicates in the form of $f\big(L(s)\big) < c$, where c is a user-specified threshold. Next, we define a *robustness function* $\rho : \mathsf{Traces}(\mathcal{Z}^*) \times \Psi \to \mathbb{R}$ that assigns a robustness value to a finite trace for an LTL formula. Intuitively, the robustness value evaluates "how far" the given finite trace is from satisfying ψ. The complete quantitative semantics is shown in Figure 3. We use constants ρ_{max} and ρ_{min} for the maximum and minimum robustness values, respectively. Given a trace, a higher ρ value implies the trace has higher robustness to satisfy ψ, and a lower ρ value means the trace is less likely to satisfy ψ (e.g., a potential violation).

Formally, given an LTL formula ψ and an MDP $\mathcal{M}$, a path ζ with a higher robustness value indicates that it has higher probability to satisfy ψ. The optimization task of seeking a single policy $\pi^\star$ is:

$$
\pi^\star \in \arg\max_{\pi} \mathop{\mathbb{P}}_{\zeta\sim\mathcal{D}_\pi}\Big[\rho\big(\mathsf{Tr}(\zeta_{[0:k]}), \psi\big) \xrightarrow{\star} \rho_{max}\Big]
$$

Here, for simplicity, we use the notation $\xrightarrow{\star}$ to represent convergence. That is, $\pi^\star$ maximizes the probability of satisfying ψ over the distribution of paths generated by policy π.

Robustness for a Tuple of Traces. To evaluate the robustness value over multiple traces, we first define a zip function that pointwise bundles a tuple of traces. Given a tuple of finite traces $\langle t_1, \ldots, t_n \rangle$, we derive a zipped trace $\mathsf{zip}(\langle t_1, \ldots, t_n \rangle)$ and for all $i \geq 0$, $\mathsf{zip}(\langle t_1, \ldots, t_n \rangle)(i) \triangleq \langle t_1(i), \ldots, t_n(i) \rangle$. Given an LTL formula, a tuple of paths $\langle \zeta_1, \zeta_2, \ldots, \zeta_n \rangle$ has higher probability of satisfying ψ if the

robustness value of $\mathsf{zip}\big(\langle \mathsf{Tr}(\zeta_{1[0:k_1]}), \mathsf{Tr}(\zeta_{2[0:k_2]}), \ldots, \mathsf{Tr}(\zeta_{n[0:k_n]})\rangle\big)$ converges to ρ_{max} w.r.t. ψ for some $k_1, \ldots, k_n$, where $0 \leq k_\ell \leq |\zeta_\ell|$ for each $1 \leq \ell \leq n$. Thus, the optimization task of computing a tuple of policies $\langle \pi_1^\star, \pi_2^\star, \ldots, \pi_n^\star \rangle$ that maximizes the robustness becomes:

$$\langle \pi_\ell^\star \rangle_{\ell \in \{1,\ldots n\}} \in \left[\arg\max_{\langle \pi_\ell \rangle} \mathbb{P}\Big[\rho\big(\mathsf{zip}\big(\langle \mathsf{Tr}(\zeta_{\ell[0:k_\ell]} \sim \mathcal{D}_{\pi_\ell})\rangle\big), \psi\big) \xrightarrow{\star} \rho_{max}\Big]\right]_{\ell \in \{1,\ldots,n\}}$$

This formulation reflects that LTL formulas are implicitly universally quantified as we emphasized in Section 1. For instance, model checking $\forall^*.\psi$ reduces to checking $\forall.\psi$ via self-composition [5, 8].

Robustness for Skolemized HyperLTL. Optimizing an alternating HyperLTL formula requires that policies for universally quantified paths simultaneously optimize the existentially quantified paths. In order to preserve ordering, we use $\cup_\leq$ to make sure the union of trace tuples are in order w.r.t. their path indices. That is, given two tuple of traces $\langle\cdot\rangle_1$ and $\langle\cdot\rangle_2$, we define $\langle\cdot\rangle_1 \cup_\leq \langle\cdot\rangle_1 \triangleq \langle \mathsf{Tr}(\zeta_x)\rangle_{x \in \{1\cdots n\}}$, where each $\mathsf{Tr}(\zeta_x)$ is either from $\langle\cdot\rangle_1$ or $\langle\cdot\rangle_2$. Given a Skolemized HyperLTL formula, the inner LTL formula is denoted as $\mathbf{Skolem}(\psi)$, as defined in (1). For each $i \in \mathbb{Q}^\exists$ and $j \in \mathbb{Q}^\forall$, we say a tuple of paths $\langle \zeta_1, \zeta_2, \ldots, \zeta_n \rangle$ satisfies ψ if and only if:

$$\Big[\rho\Big(\mathsf{zip}\big(\langle \mathsf{Tr}(\zeta_{i[0:k_i]})\rangle \cup_\leq \langle \mathsf{Tr}(\zeta_{j\,[0:k_j]})\rangle\big), \mathbf{Skolem}(\psi)\Big)\Big]_{i \in \mathbb{Q}^\exists, j \in \mathbb{Q}^\forall} \xrightarrow{\star} \rho_{max}$$

Essentially, the set $\langle \mathsf{Tr}(\zeta_{i[0:k_i]})\rangle_{i \in \mathbb{Q}^\exists}$ captures the tuple of traces produced by each Skolem function for each existential quantifiers (i.e., a trace from $Img(\mathbf{f}_i)$) for each $i \in \mathbb{Q}^\exists$), while $\langle \mathsf{Tr}(\zeta_{j\,[0:k_j]})\rangle_{j \in \mathbb{Q}^\forall}$ corresponds to the tuple of traces for each $j \in \mathbb{Q}^\forall$. That is, we can formalize the optimization task for a Skolemized HyperLTL as follows:

$$\Big[\langle \pi_i^\star \rangle \cup_\leq \langle \pi_j^\star \rangle \in \arg\max_{\langle \pi_i \rangle \cup_\leq \langle \pi_j \rangle} \mathbb{P}\Big[\rho\Big(\mathsf{zip}\big(\langle \mathsf{Tr}(\zeta_{i[0:k_i]} \sim \mathcal{D}_{\pi_i})\rangle \cup_\leq \langle \mathsf{Tr}(\zeta_{j\,[0:k_j]} \sim \mathcal{D}_{\pi_j})\rangle\big), \mathbf{Skolem}(\psi)\Big) \xrightarrow{\star} \rho_{max}\Big]\Big]_{i \in \mathbb{Q}^\exists, j \in \mathbb{Q}^\forall} \tag{2}$$

Notice that, for each $i \in \mathbb{Q}^\exists$ (i.e., a Skolem function), the evaluation of ρ dependes on whether the sampled trace $\zeta_i \sim \mathcal{D}_{\pi_i}$ *serves* as a witness for its preceding $\langle \zeta_{i_j} \rangle_{i_j \in \mathbb{Q}_i^\forall}$. That is, $\rho\big(\mathbf{f}_i, \psi\big) \triangleq \rho\big(\mathbf{f}_i(\mathsf{Tr}(\zeta_{i_1}), \ldots, \mathsf{Tr}(\zeta_{i_{|\mathbb{Q}_i^\forall|}})), \psi\big)$. Hence, the optimization of $\langle \pi_i^\star \rangle$ depends on $\langle \pi_j^\star \rangle$, which is necessary to capture the semantics of a HyperLTL formula with quantifier alternation (we provide a comprehensive example in Appendix C). Finally, we show that the set of optimal policies obtained by (2) also solves the original optimization problem in Section 3 (detailed proof in Appendix A).

Theorem 1 *Given an MDP $\mathcal{M}$ and a HyperLTL formula φ, an optimal tuple of policies $\langle \pi_i^\star \rangle_{i \in \mathbb{Q}^\exists} \cup_\leq \langle \pi_j^\star \rangle_{j \in \mathbb{Q}^\forall}$ for $\mathbf{Skolem}(\varphi)$ is also an optimal tuple of policies for φ, that optimizes the probability of satisfying φ in $\mathcal{M}$ (the problem statement in Section 3).*

5.3 Step 3: Reinforcement Learning for HyperLTL

Now, we solve the optimization task in (2) using RL, where the reward signal is defined by the robustness values from Section 5.2. As our MDP model lacks a built-in reward function, we use robustness values to guide learning. To avoid confusion with standard RL terminology, we refer to expected robustness value when mentioning expected rewards. For simplicity, we assume all sampled paths have equal length.

Optimizing Expected Reward. The goal of this step is to learn an optimal neural network $\mathcal{NN}^\star$, a parameterized compositional function that synthesizes the policies solving (2). The construction of learning constraints is inspired by the well-known *Bellman Equation* [4], which characterizes the optimal expected reward of a decision process as a recursive function of immediate reward and the value of future states. Let us use s_k to denoted the state of a zipped path at position k (i.e., $s_k \triangleq \mathsf{zip}(\langle t_1, \ldots, t_n\rangle)(k)$). We define the *immediate reward* for a state-action pair (s_k, a_k) based on the robustness value of the zipped trace from (2):

$$R(s_k, a_k) \triangleq \Big[\rho\Big(\mathsf{zip}\big(\langle \mathsf{Tr}(\zeta_{i[0:k]})\rangle \cup_\leq \langle \mathsf{Tr}(\zeta_{j\,[0:k]})\rangle\big), \mathbf{Skolem}(\psi)\Big)\Big]_{i \in \mathbb{Q}^\exists, j \in \mathbb{Q}^\forall}$$

https://doi.org/10.52202/085713-1743

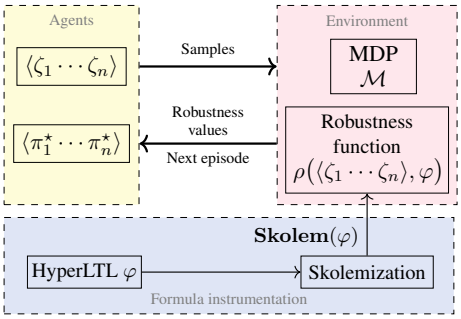


Figure 4: Overview of HYPRL.

Table 1: HyperLTL specifications of case studies.

(SRL)	$\forall\tau_1.\exists\tau_2.\ \square\langle x_{\tau_1}, y_{\tau_1}\rangle \neq \langle x_{\tau_2}, y_{\tau_2}\rangle \wedge \lozenge\langle x_{\tau_1}, y_{\tau_1}\rangle = \langle x_{G1}, y_{G1}\rangle \wedge \lozenge\langle x_{\tau_2}, y_{\tau_2}\rangle = \langle x_{G2}, y_{G2}\rangle$
(DST)	$\forall\tau_1.\exists\tau_2.\ \lozenge(\mathit{T1}_{\tau_1} \wedge \lozenge(\mathit{T2}_{\tau_1} \wedge \lozenge(\mathit{T3}_{\tau_1}) \ldots)) \wedge \square(\mathit{step}_{\tau_2} < \delta) \wedge \square(\lvert \mathit{pos}_{\tau_1} - \mathit{pos}_{\tau_2} \rvert < 1)$
(PCP)	$\forall\tau_1.\exists\tau_2.\ \psi_{\mathsf{SemiMatch}_{\tau_1}}\ \mathcal{U}\ (\psi_{\mathsf{Extend}_{\tau_1,\tau_2}} \wedge \bigwedge_{p\in\mathsf{AP}} (p_{top_{\tau_2}} \leftrightarrow p_{bot_{\tau_2}}))$ $\psi_{\mathsf{SemiMatch}_{\tau_1}} \triangleq \Big[\bigwedge_{p\in\mathsf{AP}} (p_{top_{\tau_1}} \leftrightarrow p_{bot_{\tau_1}})\Big]\ \mathcal{U}\ (\#_{top_{\tau_1}} \oplus \#_{bot_{\tau_1}})$ $\varphi_{\mathsf{Extend}_{\tau_1,\tau_2}} \triangleq \Big[\bigwedge_{p\in\mathsf{AP}} ((p_{top_{\tau_1}} \leftrightarrow p_{top_{\tau_2}}) \wedge (p_{bot_{\tau_1}} \leftrightarrow p_{bot_{\tau_2}}))\Big]\ \mathcal{U}$ $((\#_{top_{\tau_1}} \vee \#_{bot_{\tau_1}}) \wedge (\neg\#_{top_{\tau_2}} \wedge \neg\#_{bot_{\tau_2}}))$

The *expected reward* $\mathbb{E}(s_k, a_k)$ follows the classic Bellman formulation, combining immediate reward and future reward based on some discount factor γ (see Appendix B for details). Intuitively, $\mathbb{E}(s_k, a_k)$ quantifies the long-term utility of taking action a_k at state s_k. Finally, for any arbitrary state s and action a, the Bellman Equation then defines the Q-value recursively for each $(s, a) \in S \times A$, denoted as $Q^{\mathcal{NN}}(s, a)$:

$$Q^{\mathcal{NN}}(s,a) \triangleq \sum_{s'\in S} \mathbf{P}(s,a,s') \Big[R(s,a) + \gamma \sum_{a'\in A} \mathcal{NN}(a' \mid s')\, \mathbb{E}(s',a') \Big],$$

where $\mathcal{NN}(a' \mid s')$ indicates that $\mathcal{NN}$ takes action a' on state s'. Consequently, the optimal action-value function for each (s, a) is:

$$Q^{\mathcal{NN}^\star}(s,a) \triangleq \max_{\mathcal{NN}} Q^{\mathcal{NN}}(s,a) \tag{3}$$

Intuitively, $Q^{\mathcal{NN}^\star}(s, a)$ captures the maximum expected reward achievable from a state s by taking action a. We remark that, our framework samples all paths for each quantifier simultaneously. That is, the learned neural network $\mathcal{NN}^\star$ induces a set of n functions $\{\mathcal{NN}_1^\star, \ldots, \mathcal{NN}_n^\star\}$, where $n = |\mathit{Vars}(\varphi)|$, and for all $1 \leq \ell \leq n$, $\mathcal{NN}_\ell^\star$ maps a state to an optimal actions for a path ζ_ℓ.

Constructing Policies. Based on the learned $\mathcal{NN}^\star$, we now construct the tuple of policies $\langle \pi_i^\star \rangle_{i\in\mathbb{Q}^\exists}$ and $\langle \pi_j^\star \rangle_{j\in\mathbb{Q}^\forall}$ that solves (2) for each k-step ranging over the sample size as follows.

- For each $j \in \mathbb{Q}^\forall$, we inductively construct the policies:

$$\pi_j^\star(\zeta_{j\,[0:k]}) \triangleq \mathcal{NN}_j^\star(s_k)$$

- For each $i \in \mathbb{Q}^\exists$, we construct a Skolem witness as follows:

$$\pi_i^\star(\zeta_{i\,[0:k]}) \triangleq \mathcal{NN}_i^\star\Big(\mathbf{f}_i\big(\mathsf{Tr}(\zeta_{i_1\,[0:k]}), \ldots, \mathsf{Tr}(\zeta_{i_{|\mathbb{Q}_i^\forall|}\,[0:k]})\big)\Big)$$

That is, the optimal policy for a finite prefix ζ_i with $i \in \mathbb{Q}^\exists$ depends on the optimal actions taken along the preceding universal paths $\zeta_{i_1}, \ldots, \zeta_{i_{|\mathbb{Q}_i^\forall|}}$, illustrating how our framework captures the dependency of an existential path on the universally quantified ones. To this end, from the learned neural network $\mathcal{NN}^\star$, we succesfully derive two tuples $\langle \pi_i^\star \rangle_{i\in\mathbb{Q}^\exists}$ and $\langle \pi_j^\star \rangle_{j\in\mathbb{Q}^\forall}$ for Equation (2).

Theorem 2 *Given an MDP $\mathcal{M}$ and a HyperLTL formula φ, the optimal neural network function $\mathcal{NN}^\star$ derives a tuple of Skolem function witnesses $\langle \mathbf{f}_i \rangle_{i\in\mathbb{Q}^\exists}$ and a tuple of optimal policies $\langle \pi_j^\star \rangle_{j\in\mathbb{Q}^\forall}$ that optimize the satisfaction of* $\mathbf{Skolem}(\varphi)$.

Theorem 2 gives the premise of Theorem 1, which, in turn, solves the original problem stated in Section 3 (detailed proof in Appendix A).

6 Implementation and Experiments

HYPRL is fully implemented (see Figure 4). Given a HyperLTL formula φ with n quantifiers, HYPRL first constructs its Skolemized form $\mathbf{Skolem}(\varphi)$, as prescribed in Section 5.1. Next, at each step

Table 2: Descriptions of the case studies.

Safe RL in Grid Worlds (SRL)	Two agents collaboratively learn policies $\langle \pi_1^\star, \pi_2^\star \rangle$ to reach their respective goals while avoiding collisions in a grid world.
Deep Sea Treasure (DST)	Two agents (a driver and a treasure collector) navigate among treasures. Learn policies $\langle \pi_1^\star, \pi_2^\star \rangle$ such that, for all ways the collector maximizes treasures, a safe route exists for the driver to exit within the time limit δ.
Post Correspondence Problem (PCP)	Learn policies $\langle \pi_1^\star, \pi_2^\star \rangle$ for PCP: for any semi-matching sequence of dominos, there exists an extension into full matches.

t, for each $i \in \{1, \ldots, n\}$, a policy π_i observes a finite path $\zeta_{i[0:t]}$, and selects an action $\pi_i(\zeta_{i[0:t]})$, yielding an updated path $\zeta_{i[0:t+1]}$. The environment computes feedback using robustness function ρ to guide learning (Section 5.2). The optimal tuple of policies $\langle \pi_1^\star, \ldots, \pi_n^\star \rangle$ are learned iteratively using a selected RL algorithm such as DQN [36], PPO [39], or CQ-Learning [15] (Section 5.3). We summarize all our case studies and their corresponding HyperLTL specifications in Table 2 and Table 1, respectively. We also investigate the wildfire scenario introduced in Section 1 with larger grid-worlds. All experiments are ran on an Apple M1 Max (10-core CPU, 24-core GPU). Additional details of experimental setups are provided in Appendix D.

Baseline. We experiment all cases using the setup from [32] by comparing against manually designed reward functions. To ensure a fair comparison, for DST, we use the reward function from [42]; for the rest of the cases, we construct several reward functions and report the best-performing one (details are provided in Appendix D). For SRL, we also compare with the method called "shield synthesis" [17], where a pre-synthesized *shield* enforces a safety property during learning and, in this specific case, can reason about inter-agent dependencies.

Results and Analysis. First, for SRL, we use the maps from [35] and compare CQ-learning+HYPRL against shield synthesis [17] (detailed map descriptions are inAppendix D). It is worth noting that for [17], a shield has to be explicitly designed to avoid unsafe states, whereas HYPRL achieves better safe and goal-directed behavior purely through the robustness values derived from a HyperLTL formula. Our analysis for SRL is presented in Table 3. In a two-agent setting, the result shows that the policies learned by HYPRL outperform the shielded agent in all of the environments by requiring fewer steps to reach the goal, except in SUNY, where their performance is quite similar. Furthermore, in both SUNY and MIT maps, HYPRL successfully avoids collisions as what we replicated from [17]. In a three-agent setting (where the HyperLTL formula is expanded to $\forall\forall\exists$), the shielding method successfully avoids all collisions; however, it failed to reach the goals within the bound of 100 steps in ISR, Pentagon, and MIT. In contrast, HYPRL was able to guide the agents to reach their goals significantly faster than both baselines and achieved fewer collisions compared to CQ-learning. We emphasize that although shield synthesis can avoid collision in all cases, it comes with the cost of spending a significant number of steps to reach the goal. In contrast, HYPRL maintains low collision rates in ISR and Pentagon, and learns policies that reach the goal with substantially fewer steps in ISR, Pentagon, and MIT maps.

Second, for SRL, we compare DQN+HYPRL with DQN and present the results in Figure 6. The top four figures demonstrate that HYPRL always achieves a higher number of successful goal completions under the same number of learning episodes. The bottom four figures show that HYPRL requires fewer episodes to learn collision avoidance.

For DST, we compare PPO+HYPRL with PPO, and DQN+HYPRL with DQN, using the reward functions from [42] in both cases. We evaluate using two metrics: (1) the total amount of collected treasures ($\sum$ ), and (2) the average amount of collected treasures per step ($\sum$ $/steps$). As shown

Table 3: SRL, avg. and std. error of 10 evaluations after 1k training episodes, 10 runs, where S/B stands for step-bound which is 100 steps for all cases.

		CQ		CQ + Shield [17]		CQ + HYPRL	
Maps	No. Agents	Steps	Collisions	Steps	Collisions	Steps	Collisions
ISR	2	27.95±7.4	0.19±0.1	17.40±2.2	**0.00**±0.0	**7.58**±0.3	0.25±0.2
Pentagon	2	36.46±7.7	0.28±0.1	75.20±12.6	**0.00**±0.0	**11.90**±4.6	0.53±0.5
SUNY	2	11.99±0.5	0.01±0.0	**11.50**±0.3	**0.00**±0.0	12.48±0.6	**0.00**±0.0
MIT	2	41.28±8.5	0.20±0.1	33.46±3.4	**0.00**±0.0	**23.20**±0.5	**0.00**±0.0
ISR	3	98.79±0.8	12.68±3.8	S/B	**0.00**±0.0	**74.18**±5.1	7.78±1.0
Pentagon	3	97.15±2.4	16.46±7.2	S/B	**0.00**±0.0	**78.82**±1.7	10.92±1.4
SUNY	3	84.89±7.9	0.63±0.2	82.35±4.1	**0.00**±0.0	**44.95**±8.3	0.71±0.4
MIT	3	96.96±1.8	2.83±1.3	S/B	**0.00**±0.0	**71.53**±7.7	1.58±0.7

https://doi.org/10.52202/085713-1743

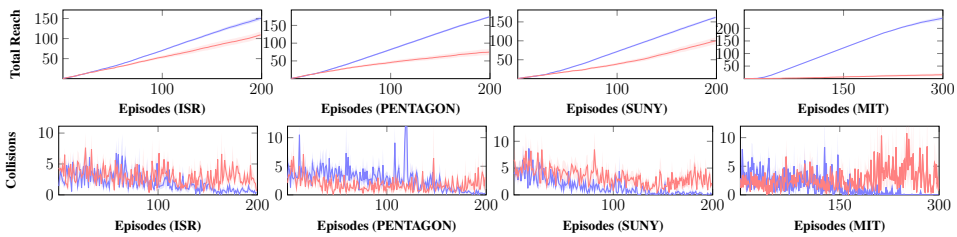


Figure 6: SRL results, avg. and std. error of total number of goal completions by 2 agents (top) and collisions (bottom). DQN+HYPRL (line ——) and DQN (line ——), over 10 runs.

Table 4: DST results, avg. and std. error over 10 evaluations across, 10 runs, and Epi. stands for "Episodes".

Method	Epi.	∑	∑/*steps*	Method	Epi.	∑	∑/*steps*
PPO	500	4.31±1.2	0.17±0.04	DQN	500	1.08±0.2	0.05±0.00
PPO + HYPRL	500	**22.93**±2.2	**0.91**±0.08	DQN + HYPRL	500	**4.12**±1.4	**0.14**±0.05
PPO	1000	3.22±0.8	0.12±0.03	DQN	1000	1.45±0.2	0.02±0.00
PPO + HYPRL	1000	**23.97**±2.2	**1.12**±0.08	DQN + HYPRL	1000	**4.43**±0.8	**0.21**±0.03

Table 5: Wildfire results, avg. and std. error of 10 trials after 5k episodes, 10 runs.

Size	Method	Dist	Steps O_1	Steps O_2
3^2	PPO	2.5±0.01	33.43±4.1	787.03±31.8
	PPO + HYPRL	**2.30**±0.03	**18.940**±1.1	**143.550**±1.1
5^2	PPO	4.2±0.01	62.7±9.7	S/B
	PPO + HYPRL	**2.1**±0.05	**59.50**±14.3	**8057.5**±121.4
8^2	PPO	11.2±0.03	16801.8±2144.0	S/B
	PPO + HYPRL	**6.94**±0.07	**4149.6**±1743.1	**386.2**±80.5
10^2	PPO	10.9±0.01	S/B	29023.6±976.4
	PPO + HYPRL	**5.3**±0.10	**21272.8**±3579.0	**570.3**±52.2

in Table 4, policies learned by HYPRL outperform those learned by the baseline, regardless of the number of training episodes or the choice of the RL algorithm.

Finding policies to solve the PCP problem is inherently challenging due to its undecidability. HYPRL is able to learn optimal policies that solve PCP more efficient, compare to the baseline. Figure 5 presents the performance comparison of DQN+HYPRL with DQN, showing that HYPRL achieves more successful matches in both 5- and 6-domino settings.

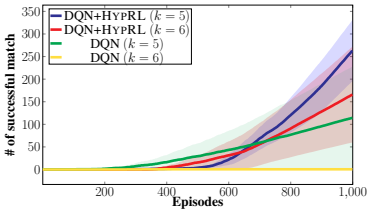


Figure 5: PCP results, avg. and std. error of total matches, 10 runs.

Lastly, we evaluate HYPRL on the wildfire scenario and compare PPO+HYPRL with PPO. As shown in Table 5, even when scaling up the grid-world environment from 3^2 to 5^2, 8^2, and 10^2, HYPRL consistently outperforms the baseline by requiring fewer steps to rescue victims, extinguish all fires, and maintain a safe distance between the two agents. The step-bounds are set to 10k steps for the 3^2 and 5^2 environments, 20k steps for 8^2, and 30k steps for 10^2. In fact, PPO fails to learn policies capable of rescuing victims (O_2) within the step-bounds for 5^2 and 8^2, and is unable to extinguish all fires (O_1) within the step-bound in the 10^2 setting.

7 Conclusion

We introduced HYPRL, a reinforcement learning framework that synthesizes control policies for tasks specified as HyperLTL formulas over MDPs with unknown transitions. By leveraging the expressiveness of hyperproperties, HYPRL can handle complex multi-agent objectives and relational constraints, including safe planning and undecidable problems like PCP. Our quantitative semantics automatically derives robustness values from the specification and learns control policies that maximize the probability of satisfying the given HyperLTL property. Implemented with off-the-shelf RL algorithms, HYPRL demonstrates superior performance over comparable baselines across a set of diverse case studies, especially in scenarios where reward shaping is non-trivial.

Limitations. HYPRL currently does not support fully decentralized control policies, where each agent learns its local policy without access to global observations and rewards. Second, it is possible that environment states are not fully observable. That is, instead of an MDP, the environment is a partially observable MDP (POMDP). We believe these limitations open up exciting future work and can be addressed by extending HYPRL.

Societal impacts. This work marks a significant step towards extending MARL to complex tasks with relational requirements. By enabling the learning of policies under interdependent objectives, our approach broadens the applicability of MARL to critical domains such as disaster response, swarm drones, smart home automation, and industrial resource management, where coordination among agents is essential for success.

https://doi.org/10.52202/085713-1743

References

[1] Alekh Agarwal, Nan Jiang, Sham M Kakade, and Wen Sun. Reinforcement learning: Theory and algorithms. *CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep*, 32:96, 2019.

[2] Derya Aksaray, Austin Jones, Zhaodan Kong, Mac Schwager, and Calin Belta. Q-learning for robust satisfaction of signal temporal logic specifications. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 6565–6570. IEEE, 2016.

[3] Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. Safe reinforcement learning via shielding. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[4] EN Barron and H Ishii. The bellman equation for minimizing the maximum cost. *NONLINEAR ANAL. THEORY METHODS APPLIC.*, 13(9):1067–1090, 1989.

[5] Gilles Barthe, Pedro R D'argenio, and Tamara Rezk. Secure information flow by self-composition. *Mathematical Structures in Computer Science*, 21(6):1207–1252, 2011.

[6] Richard Bellman. On the theory of dynamic programming. *Proceedings of the national Academy of Sciences*, 38(8):716–719, 1952.

[7] Raven Beutner and Bernd Finkbeiner. Hyperatl*: A logic for hyperproperties in multi-agent systems. *Logical Methods in Computer Science*, Volume 19, Issue 2:13, May 2023. ISSN 1860-5974.

[8] B. Bonakdarpour and B. Finkbeiner. The complexity of monitoring hyperproperties. In *Proceedings of the 31st IEEE Computer Security Foundations Symposium CSF*, pages 162–174, 2018.

[9] B. Bonakdarpour and S. Sheinvald. Finite-word hyperlanguages. *Information and Computation*, 295:104944, 2023.

[10] Ronen Brafman, Giuseppe De Giacomo, and Fabio Patrizi. Ltlf/ldlf non-markovian rewards. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[11] N. Brett, U. Siddique, and B. Bonakdarpour. Rewriting-based runtime verification for alternation-free HyperLTL. In *Proceedings of the 23rd International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, pages 77–93, 2017.

[12] M. R. Clarkson and F. B. Schneider. Hyperproperties. *Journal of Computer Security*, 18(6): 1157–1210, 2010.

[13] M. R. Clarkson, B. Finkbeiner, M. Koleini, K. K. Micinski, M. N. Rabe, and C. Sánchez. Temporal logics for hyperproperties. In *Proceedings of the 3rd Conference on Principles of Security and Trust POST*, pages 265–284, 2014.

[14] Giuseppe De Giacomo, Luca Iocchi, Marco Favorito, and Fabio Patrizi. Foundations for restraining bolts: Reinforcement learning with ltlf/ldlf restraining specifications. In *Proceedings of the international conference on automated planning and scheduling*, volume 29, pages 128–136, 2019.

[15] Yann-Michaël De Hauwere, Peter Vrancx, and Ann Nowé. Learning multi-agent state space representations. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: Volume 1 - Volume 1*, AAMAS '10, page 715–722, Richland, SC, 2010. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9780982657119.

[16] Ingy ElSayed-Aly and Lu Feng. Logic-based reward shaping for multi-agent reinforcement learning, 2022.

[17] Ingy ElSayed-Aly, Suda Bharadwaj, Christopher Amato, Rüdiger Ehlers, Ufuk Topcu, and Lu Feng. Safe multi-agent reinforcement learning via shielding. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '21, page 483–491, Richland, SC, 2021. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450383073.

 https://doi.org/10.52202/085713-1743

[18] Ky Fan. Minimax theorems. *Proceedings of the National Academy of Sciences*, 39(1):42–47, 1953.

[19] B. Finkbeiner and C. Hahn. Deciding hyperproperties. In *Proceedings of the 27th International Conference on Concurrency Theory (CONCUR)*, pages 13:1–13:14, 2016.

[20] Lewis Hammond, Alessandro Abate, Julian Gutierrez, and Michael Wooldridge. Multi-agent reinforcement learning with temporal logic specifications, 2021.

[21] Mohammadhosein Hasanbeig, Alessandro Abate, and Daniel Kroening. Logically-constrained reinforcement learning. *arXiv preprint arXiv:1801.08099*, 2018.

[22] Mohammadhosein Hasanbeig, Yiannis Kantaros, Alessandro Abate, Daniel Kroening, George J Pappas, and Insup Lee. Reinforcement learning for temporal logic control synthesis with probabilistic satisfaction guarantees. In *2019 IEEE 58th conference on decision and control (CDC)*, pages 5338–5343. IEEE, 2019.

[23] Kishor Jothimurugan, Rajeev Alur, and Osbert Bastani. A composable specification language for reinforcement learning tasks. *Advances in Neural Information Processing Systems*, 32, 2019.

[24] Kishor Jothimurugan, Suguman Bansal, Osbert Bastani, and Rajeev Alur. Compositional reinforcement learning from logical specifications. *Advances in Neural Information Processing Systems*, 34:10026–10039, 2021.

[25] Kishor Jothimurugan, Suguman Bansal, Osbert Bastani, and Rajeev Alur. Specification-guided learning of nash equilibria with high social welfare. In *International Conference on Computer Aided Verification*, pages 343–363. Springer, 2022.

[26] Bettina Könighofer, Florian Lorber, Nils Jansen, and Roderick Bloem. Shield synthesis for reinforcement learning. In *Leveraging Applications of Formal Methods, Verification and Validation: Verification Principles: 9th International Symposium on Leveraging Applications of Formal Methods, ISoLA 2020, Rhodes, Greece, October 20–30, 2020, Proceedings, Part I 9*, pages 290–306. Springer, 2020.

[27] Bettina Könighofer, Julian Rudolf, Alexander Palmisano, Martin Tappler, and Roderick Bloem. Online shielding for reinforcement learning. *Innovations in Systems and Software Engineering*, 19(4):379–394, 2023.

[28] Yen-Ling Kuo, Boris Katz, and Andrei Barbu. Encoding formulas as deep networks: Reinforcement learning for zero-shot execution of ltl formulas. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5604–5610. IEEE, 2020.

[29] Orna Kupferman and Moshe Y. Vardi. Model checking of safety properties. In *International Conference on Computer Aided Verification, CAV 1999*, 1999.

[30] Xuan-Bach Le, Dominik Wagner, Leon Witzman, Alexander Rabinovich, and Luke Ong. Reinforcement learning with LTL and ω-regular objectives via optimality-preserving translation to average rewards. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[31] Borja G. León and Francesco Belardinelli. Extended markov games to learn multiple tasks in multi-agent reinforcement learning, 2020.

[32] Xiao Li, Cristian-Ioan Vasile, and Calin Belta. Reinforcement learning with temporal logic rewards. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3834–3839. IEEE, 2017.

[33] Chanjuan Liu, Jinmiao Cong, Bingcai Chen, Yaochu Jin, and Enqiang Zhu. Guiding multi-agent multi-task reinforcement learning by a hierarchical framework with logical reward shaping, 2024.

[34] Daniel Melcer, Christopher Amato, and Stavros Tripakis. Shield decentralization for safe multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 35: 13367–13379, 2022.

[35] Francisco S. Melo and Manuela Veloso. Learning of coordination: exploiting sparse interactions in multiagent systems. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '09, page 773–780, Richland, SC, 2009. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9780981738178.

[36] 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, February 2015. ISSN 1476-4687.

[37] A. Pnueli. The temporal logic of programs. In *Symposium on Foundations of Computer Science (FOCS)*, pages 46–57, 1977.

[38] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

[39] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[40] Thoralf Skolem. Logisch-kombinatorische untersuchungen über die erfüllbarkeit oder beweisbarkeit mathematischer sätze nebst einem theoreme über dichte mengen. *Videnskapsselskapets Skrifter, I. Matematisk-naturvidenskabelig Klasse*, 1920.

[41] Wolfgang Thomas. Safety-and liveness-properties in propositional temporal logic: characterizations and decidability. *Banach Center Publications*, 21(1):403–417, 1988.

[42] Peter Vamplew, Richard Dazeley, Adam Berry, Rustam Issabekov, and Evan Dekker. Empirical evaluation methods for multiobjective reinforcement learning algorithms. *Machine Learning*, 84(1):51–80, 2011.

[43] S. Winter and M. Zimmermann. Tracy, traces, and transducers: Computable counterexamples and explanations for hyperltl model-checking, 2024.

[44] Z. Xu, Y. Rawat, Y. Wong, M. S. Kankanhalli, and M. Shah. Don't pour cereal into coffee: Differentiable temporal logic for temporal action segmentation. In *Advances in Neural Information Processing Systems*, volume 35, pages 14890–14903, 2022.

[45] Zhe Xu and Ufuk Topcu. Transfer of temporal logic formulas in reinforcement learning. In *IJCAI: proceedings of the conference*, volume 28, page 4010. NIH Public Access, 2019.

[46] Runzhe Yang, Xingyuan Sun, and Karthik Narasimhan. A generalized algorithm for multi-objective reinforcement learning and policy adaptation. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[47] Lim Zun Yuan, Mohammadhosein Hasanbeig, Alessandro Abate, and Daniel Kroening. Modular deep reinforcement learning with temporal logic specifications. *arXiv e-prints*, pages arXiv–1909, 2019.

 https://doi.org/10.52202/085713-1743

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The paper successfully makes the four contributions as we stated explicitly in the abstract and the introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Yes, we elaborate the limitation in Section 7.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: We provide complete proofs in the supplementary material for the both theorem 1 and theorem 2 stated in the main paper.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We detail the steps for reproducing our main experimental results in Section 5.1,Section 5.2, and Section 5.3, and describe the implementation of our framework in Section 6. Additional implementation and experimental details are provided in the supplementary material.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

https://doi.org/10.52202/085713-1743

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: Our implementation can be reproduced by following the detailed algorithmic steps in Section 5 and the elaborations of experimental setups for each case in the Appendix.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide all details about hyperparameters and environment setup in supplementary material.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Standard error is reported in all figures and tables throughout the paper (represented by the color-shaded areas).

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: All experiments are conducted on an Apple M1 Max (10-core CPU, 24-core GPU).

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We believe we fulfill all specifications in the ethics guidelines.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Yes, we discussed the societal impacts in Section 7.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

https://doi.org/10.52202/085713-1743

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This paper poses no such risk.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We did use existing assets, and we properly cited them in the paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA] .

    Justification: This paper did not release any new assets, as the main contribution is a new framework that leverage on existing learning techniques to solve our target objective.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

https://doi.org/10.52202/085713-1743

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer:[NA]

    Justification: The core method development (i.e., technical content) in this paper does not involve LLMs.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

A Proofs of Theorems

A.1 Proof of Theorem 1

Recall that given a HyperLTL formula φ, $\mathbf{Skolem}(\varphi)$ is in the form of:

$$\mathbf{Skolem}(\varphi) = \underbrace{\exists \mathbf{f}_i(\tau_{i_1}, \ldots, \tau_{i_{|\mathbb{Q}_i^\forall|}}).}_{\text{for each } i \in \mathbb{Q}^\exists} \quad \underbrace{\forall \tau_j.}_{\text{for each } j \in \mathbb{Q}^\forall} \quad \mathbf{Skolem}(\psi)$$

Assuming that $\langle \pi_i^\star \rangle_{i \in \mathbb{Q}^\exists} \cup_\leq \langle \pi_j^\star \rangle_{j \in \mathbb{Q}^\forall}$ is the optimal set of policies for the Skolemized φ, then for every $k \geq 0$, it maximizes the probability of converging to ρ_{max} for the inner LTL sub-formula $\mathbf{Skolem}(\psi)$:

$$\mathbb{P}\Big[\rho\Big(\mathsf{zip}\big(\langle \mathsf{Tr}(\zeta_{i\,[0:k_i]})\rangle \cup_\leq \langle \mathsf{Tr}(\zeta_{j\,[0:k_j]})\rangle\big), \mathbf{Skolem}(\psi)\Big)\Big]_{i \in \mathbb{Q}^\exists, j \in \mathbb{Q}^\forall} \xrightarrow{\star} \rho_{max}$$

That is, on any step k, the zipped path derives a set of paths that instantiate each path variables in the original φ as follows:

- For all universal quantified paths τ_j, where $j \in \mathbb{Q}^\forall$:

$$[\tau_j \mapsto \mathsf{Tr}(\zeta_{j\,[0:k]})]$$

- For all existential quantified paths τ_i, where $i \in \mathbb{Q}^\exists$:

$$[\tau_i \mapsto \mathbf{f}_i\big(\mathsf{Tr}(\zeta_{i_1\,[0:k]}), \ldots, \mathsf{Tr}(\zeta_{i_{|\mathbb{Q}_i^\forall|}[0:k]})\big)], \text{ and}$$

$$\text{for each } \ell \in \{1, \ldots, |\mathbb{Q}_i^\forall|\}, \text{ if } i_\ell = j \text{ then } \mathsf{Tr}(\zeta_{i_\ell\,[0:k]}) = \mathsf{Tr}(\zeta_{j\,[0:k]})$$

This instaintiation guarantees that the probability of satisfying φ is maximized up to step k. Next, the zipped path proceeds to step $k+1$ with an optimal action given by its corresponding optimal policy $\langle \pi_i^\star \rangle_{i \in \mathbb{Q}^\exists} \cup_\leq \langle \pi_j^\star \rangle_{j \in \mathbb{Q}^\forall}$. That is, the same path mappings of each universally quantified path variable ζ_j and existentially quantified Skolem function witnesses $\mathbf{f}_i$ holds for all $k \geq 0$. Finally, for the original HyperLTL formula $\varphi = \mathbb{Q}_1\tau_1.\mathbb{Q}_2\tau_2.\ldots\mathbb{Q}_n.\tau_n.\ \psi$, by instantiating the trace variables in the same fashion for all τ_i of $i \in \mathbb{Q}^\exists$ and all τ_j of $j \in \mathbb{Q}^\forall$, the same optimality immediately follows, because the optimal convergence to ρ_{max} means maximizing the probability of satisfying the inner LTL sub-formula. That is, the optimal set of paths (derived from the zipped path) also optimizes the satisfaction of original φ for all steps $k \geq 0$. To this end, we proved that, an optimal tuple of policies $\langle \pi_i^\star \rangle_{i \in \mathbb{Q}^\exists} \cup_\leq \langle \pi_j^\star \rangle_{j \in \mathbb{Q}^\forall}$ for $\mathbf{Skolem}(\varphi)$ is also an optimal set of policies for φ, that optimizes the probability of satisfying φ in $\mathcal{M}$.

A.2 Proof of Theorem 2

Bellman's Principle of Optimality [6] states that "*for an optimal policy, no matter what the initial decision is, the remaining decisions must constitute an optimal policy with regard to the state resulting from the initial decision*". An extended lemma (proved in [1]) states that for a discounted MDP, there exists an *optimal* policy, denoted as $\mathcal{NN}^\star$, such that for all $(s, a) \in S \times A$, there exists a maximum Q-value achieved by $\mathcal{NN}^\star$ (denoted as $Q^{\mathcal{NN}^\star}$) as introduced in Equation (3):

$$Q^{\mathcal{NN}^\star}(s, a) \triangleq \max_{\mathcal{NN}} Q^{\mathcal{NN}}(s, a)$$

Let us denote $\mathcal{NN}(a \mid s)$ as $\mathcal{NN}$'s decision to take action a on state s. Our goal is to find an optimized $\mathcal{NN}$, such that:

$$\max_{\mathcal{NN}} \sum_{s' \in S} \mathbf{P}(s, a, s') \Big[R(s, a) + \gamma \sum_{a \in A} \mathcal{NN}(a \mid s) \mathbb{E}(s', a') \Big]$$

where $\mathbf{P}(s, a, s')$ is the one-step transition probability, $R(s, a)$ is the reward of taking action a on state s, γ is a discount selected factor, and $\mathbb{E}$ is the expected reward of keep taking an action a' on a state s'. (as defined in Section 5.3). Recall that in Section 5.2, our immediate reward $R(s, a)$ is associated with a robustness value of a finite prefix by evaluation only up to the current seen state

https://doi.org/10.52202/085713-1743

s (i.e., independent from the unseen s' after taking action a). As a result, maximizing $\mathcal{NN}$ do not depend on $R(s, a)$, so the previous optimization problem is equivalent to:

$$R(s,a) + \max_{\mathcal{NN}} \Big[\sum_{s' \in S} \mathbf{P}(s,a,s')\, \gamma \sum_{a \in A} \mathcal{NN}(a \mid s) \mathbb{E}(s', a') \Big]$$

Let us now only focus on the optimization part (i.e., the right side of the plus operator in the above formula). Based on the definition of expected reward defined in Section 5.3, the above formula shows that an optimal $\mathcal{NN}^\star$ is more likely to the take action a (by the learned $\mathcal{NN}(a \mid s)$) that has higher probability (decided by $\mathbf{P}(s, a, s')$) to transit to an unseen state s' that lead to higher expected value (estimated by $\mathbb{E}(s', a')$). That is, given a state s, $\mathcal{NN}^\star$ outputs an optimal action a, such that:

$$\gamma \mathbb{E}_{s' \sim \mathbf{P}(s,a,\cdot))} \big[R(s', a') \big]^{\mathcal{NN}^\star},$$

which, intuitively, represents an optimal *one-step* look-ahead. Now, to connect the above formula with the reward function defined in Section 5.3, we have:

$$\gamma \mathbb{E}_{s' \sim \mathbf{P}(s,a,\cdot))} \Big[\rho\Big(\mathsf{zip}\big(\langle \mathsf{Tr}(\zeta_{i_{[0:k]}}) \rangle \cup_{\leq} \langle \mathsf{Tr}(\zeta_{j_{[0:k]}}) \rangle \big), \mathbf{Skolem}(\psi) \Big) \Big]^{\mathcal{NN}\star}_{i \in \mathbb{Q}^\exists, j \in \mathbb{Q}^\forall},$$

where s is the state on the k-th step of the zipped path. Recall that ρ is constructed using min-max approach as presented in Figure 3, so the optimal outcome of ρ can be derived from the *Minimax Lemma* [18]. That is, if each path always considers the "worst-possible" scenario that other paths will act during learning, it leads to a set of optimal policy among all paths. Hence, ρ is guaranteed optimal for all steps $k \geq 0$, which implies the maximum probability of the following:

$$\mathbb{P}\Big[\rho\Big(\mathsf{zip}\big(\langle \mathsf{Tr}(\zeta_{i_{[0:k_i]}} \sim \mathcal{D}_{\pi_i}) \rangle \cup_{\leq} \langle \mathsf{Tr}(\zeta_{j_{[0:k_j]}} \sim \mathcal{D}_{\pi_j}) \rangle \big), \mathbf{Skolem}(\psi) \Big) \xrightarrow{\star} \rho_{max} \Big]_{i \in \mathbb{Q}^\exists, j \in \mathbb{Q}^\forall}$$

To this end, we prove that the action chose by $\mathcal{NN}^\star$ achieves the maximum expected value $\mathbb{E}$ for all $(s, a) \in S \times A$. Finally, $\langle \mathbf{f}_i \rangle_{i \in \mathbb{Q}^\exists}$ and $\langle \pi^\star_j \rangle_{j \in \mathbb{Q}^\forall}$ can be inductively constructed from $\mathcal{NN}^\star$ (as we elaborated in Section 5.3), which is a policies set that optimizes the satisfaction of $\mathbf{Skolem}(\varphi)$.

B Additional Algorithmic Details of HypRL

Expected Reward of a Zipped Trace. The *expected reward* of a state s_k after taking an action a_k with *discount factor* $\gamma \in [0, 1]$ is:

$$\mathbb{E}(s_k, a_k) \triangleq \sum_{t=0}^{\infty} \gamma^t \times R(s_{k+t+1}, a_{k+t+1})$$

Intuitively, $\mathbb{E}(s_k, a_k)$ evaluates "how good" of choosing a_k on s_k in infinite time steps, where each step is *discounted* by γ. We address that, γ is often chosen based on the optimization goal. For example, for short-term tasks, a lower γ is preferred because the expected robustness value focuses more on immediate ρ value. We report the setup of these hyperparameters in Appendix D.

C A Comprehensive Example of HypRL Algorithm

In this section, we present a comprehensive head-to-toe example that aligns with the algorithmic details presented in Section 5.

Formula Skolemization. In Section 3, we presented the HyperLTL formula for our wildfire example as follows:

$$\textit{Requirements}: \ \varphi_{\mathsf{Rescue}} \triangleq \forall \tau_1 . \exists \tau_2 . (\psi_{\mathsf{fire}} \wedge \psi_{\mathsf{save}} \wedge \psi_{\mathsf{dist}} \wedge \psi_{\mathsf{safe}})$$

$$O_1: \ \psi_{\mathsf{fire}} \triangleq \Diamond(i_{\tau_1}) \wedge \Diamond(f_{\tau_1}) \wedge \Diamond(c_{\tau_1}) \qquad C_1: \ \psi_{\mathsf{dist}} \triangleq \Box(|\mathsf{Location}_{\tau_1} - \mathsf{Location}_{\tau_2}| < 3)$$

$$O_2: \ \psi_{\mathsf{save}} \triangleq \Diamond(g_{\tau_2}) \wedge \Diamond(f_{\tau_2}) \qquad C_2: \ \psi_{\mathsf{safe}} \triangleq (\neg i_{\tau_2} \,\mathcal{U}\, i_{\tau_1}) \wedge (\neg f_{\tau_2} \,\mathcal{U}\, f_{\tau_1}) \wedge (\neg c_{\tau_2} \,\mathcal{U}\, c_{\tau_1})$$

Following the illustration in Section 5.1, the Skolemized form of the formula $\varphi_{\mathsf{Rescue}}$ is as follows:

$$\begin{aligned}\mathbf{Skolem}(\varphi_{\mathsf{Rescue}}) \triangleq \exists \mathbf{f}_2(\tau_1).\forall \tau_1.\\ \mathbf{Skolem}(\psi_{\mathsf{fire}}) \wedge \mathbf{Skolem}(\psi_{\mathsf{save}}) \wedge \mathbf{Skolem}(\psi_{\mathsf{dist}}) \wedge \mathbf{Skolem}(\psi_{\mathsf{safe}})\end{aligned}$$

$$\begin{aligned}\mathbf{Skolem}(\psi_{\mathsf{fire}}) &\triangleq \Diamond(i_{\tau_1}) \wedge \Diamond(f_{\tau_1}) \wedge \Diamond(c_{\tau_1})\\ \mathbf{Skolem}(\psi_{\mathsf{save}}) &\triangleq \Diamond(g_{\mathbf{f}_2}) \wedge \Diamond(f_{\mathbf{f}_2})\\ \mathbf{Skolem}(\psi_{\mathsf{dist}}) &\triangleq \Box(|\mathsf{Location}_{\tau_1} - \mathsf{Location}_{\mathbf{f}_2}| < 3)\\ \mathbf{Skolem}(\psi_{\mathsf{safe}}) &\triangleq (\neg i_{\mathbf{f}_2}\, \mathcal{U}\, i_{\tau_1}) \wedge (\neg f_{\mathbf{f}_2}\, \mathcal{U}\, f_{\tau_1})(\neg c_{\mathbf{f}_2}\, \mathcal{U}\, c_{\tau_1})\end{aligned}$$

In this context, there are two quantifiers, $\mathbb{Q}_2 = \exists$ so $\mathbb{Q}^{\exists} = \{2\}$, and $\mathbb{Q}_1 = \forall$ so $\mathbb{Q}^{\forall} = \{1\}$. Besides, all propositions of the existential quantified path variable (originally subscripted by τ_2) average now subscripted by $\mathbf{f}_2$ in the Skolemized form. Furthermore, for each inner LTL sub-formula, the propositions subscripted with τ_2 is substituted with propositions subscripted with $\mathbf{f}_2$.

Robustness Optimization. Continuing with the Skolemized formula:

$$\exists \mathbf{f}_2(\tau_1).\forall \tau_1.\ \mathbf{Skolem}(\psi_{\mathsf{fire}}) \wedge \mathbf{Skolem}(\psi_{\mathsf{save}}) \wedge \mathbf{Skolem}(\psi_{\mathsf{dist}}) \wedge \mathbf{Skolem}(\psi_{\mathsf{safe}})$$

Our goal is to optimize the following:

$$\begin{aligned}\langle \pi_1^\star, \pi_2^\star \rangle \in \underset{\langle \pi_1, \pi_2 \rangle}{\arg\max}\, \mathbb{P}\Big[\rho\big(\mathsf{zip}(\langle \mathsf{Tr}(\zeta_{1\,[0:k_1]} \sim \mathcal{D}_{\pi_1}), \mathsf{Tr}(\zeta_{2\,[0:k_2]} \sim \mathcal{D}_{\pi_2})\rangle),\\ \mathbf{Skolem}(\psi_{\mathsf{fire}} \wedge \psi_{\mathsf{save}} \wedge \psi_{\mathsf{dist}} \wedge \psi_{\mathsf{safe}})\big) \xrightarrow{\star} \rho_{max}\Big]\end{aligned}$$

Notice that, by applying $\cup_{\leq}$, the order of the quantified traces are in the right order with respect to the original HyperLTL formula φ.

D Implementation and Experimental Details

D.1 From HyperLTL Formulas to Robustness Functions

In our implementation, we use the theory developed in Section 5.1 to obtain the Skolemized form of the formula, denoted $\mathbf{Skolem}(\varphi)$. We then generate the corresponding robustness function using the approach presented in Section 5.2 (see [32] for details on translating temporal properties into robustness functions). We now elaborate the following term from the quantitative semantics introduced in Figure 3:

$$\rho\big(\mathsf{Tr}(\zeta_{[\ell:k]}), f\big(L(s_\ell) < c\big)\big) = c - f\big(L(s_\ell)\big)$$

This equation lies at the core of transitioning from logical properties (which yield Boolean outputs) to robustness functions (which produce real-valued outputs). To apply this transformation, we must define the function f and the constant c for each property considered in this paper. We now provide details explanations on how we define such functions and constants for each formula introduced in this paper.

- The HyperLTL formula defined for safe reinforcement learning (SRL), denoted as φ_{SRL}:
  - For the property $\langle x_{\tau_1}, y_{\tau_1} \rangle = \langle x_{G1}, y_{G1} \rangle$, we define f as a distance function and set $c = 1$. The distance function we have in this case study follows the well-known Manhattan distance definition. That is:

    $$\texttt{Dist}(\langle x_{\tau_1}, y_{\tau_1} \rangle, \langle x_{G1}, y_{G1} \rangle) \triangleq (|x_{\tau_1} - x_{G1}| + |y_{\tau_1} - y_{G1}|)$$

    The, by setting $c = 1$, this yields the following expression:

    $$\texttt{Dist}(\langle x_{\tau_1}, y_{\tau_1} \rangle, \langle x_{G1}, y_{G1} \rangle) < 1,$$

    which expresses that the distance from the first agent (i.e., τ_1) to its goal (i.e., G_1) is less than one, which implies that the first agent has *indeed* reached the specified goal location. The corresponding equation is:

    $$1 - \texttt{Dist}(\langle x_{\tau_1}, y_{\tau_1} \rangle, \langle x_{G1}, y_{G1} \rangle)$$

    The optimal value is achieved when the agent is exactly at the location of the goal position, giving $1 - 0 = 1$.

https://doi.org/10.52202/085713-1743

– For the property $\langle x_{\tau_2}, y_{\tau_2}\rangle = \langle x_{G2}, y_{G2}\rangle$, the same definition for the distance function is similarly defined. That is:

$$\texttt{Dist}(\langle x_{\tau_2}, y_{\tau_2}\rangle, \langle x_{G2}, y_{G2}\rangle) < 1,$$

which expresses that the distance from the second agent to its goal is less than one, so the corresponding equation is:

$$1 - \texttt{Dist}(\langle x_{\tau_2}, y_{\tau_2}\rangle, \langle x_{G2}, y_{G2}\rangle)$$

The optimal value is achieved when the second agent is exactly at the goal position, giving $1 - 0 = 1$.

– For the property $\langle x_{\tau_1}, y_{\tau_1}\rangle \neq \langle x_{\tau_2}, y_{\tau_2}\rangle$, we define f as a distance function and set $c = -1$. This yields the condition:

$$-\texttt{Dist}(\langle x_{\tau_1}, y_{\tau_1}\rangle, \langle x_{\tau_2}, y_{\tau_2}\rangle) < -1,$$

which expresses that the distance between the first and second agent (i.e., τ_1 and τ_2) is greater than one; i.e., the agents are not in the same location. The corresponding equation is:

$$-1 + \texttt{Dist}(\langle x_{\tau_1}, y_{\tau_1}\rangle, \langle x_{\tau_2}, y_{\tau_2}\rangle).$$

This equation is maximized when the two agents are far apart, and it evaluates to -1 when they are in same location.

- The HyperLTL formula defined for deep sea treasure (DST), denoted as φ_{DST}:
  – For the properties *T1*, *T2*, $\cdots$, we define f for each treasure *Ti* a distance function between the position of the submarine driver (τ_i) and the treasure location (*Ti*), and set $c = 1$. This yields the condition:

$$\texttt{Dist}(\langle x_{\tau_1}, y_{\tau_1}\rangle, \langle x_{T1}, y_{T1}\rangle) < 1$$

which expresses that the distance from the first agent to *T1* is less than one. The corresponding equation is:

$$1 - \texttt{Dist}(\langle x_{\tau_1}, y_{\tau_1}\rangle, \langle x_{T1}, y_{T1}\rangle)$$

The same approach is applied to *T2*, *T3*,... and the remaining properties.

  – For the property $step_{\tau_2} < \delta$, we use the same approach and obtain the following equation:

$$\delta - step_{\tau_2}$$

  – For the property $|pos_{\tau_1} - pos_{\tau_2}| < 1$, we define f as a distance function and set $c = 1$. This yields the condition:

$$\texttt{Dist}(\langle x_{\tau_1}, y_{\tau_1}\rangle, \langle x_{\tau_2}, y_{\tau_2}\rangle) < 1,$$

which expresses that the distance between the first and second agent is less than one. That is, the agents are in the same location. The corresponding equation is:

$$1 - \texttt{Dist}(\langle x_{\tau_1}, y_{\tau_1}\rangle, \langle x_{\tau_2}, y_{\tau_2}\rangle)$$

This equation is maximized when the agents are in the same location.

- The formula defined for the Post Correspondence Problem (PCP), denoted as φ_{PCP}:
  – To address the properties such as $(p_1 \leftrightarrow p_2)$, where p means a sequence of alphabets (that forms a string), we define f as a function of evaluating whether two strings are matching and set $c = 1$. The matching function we have in this case study is as follows:

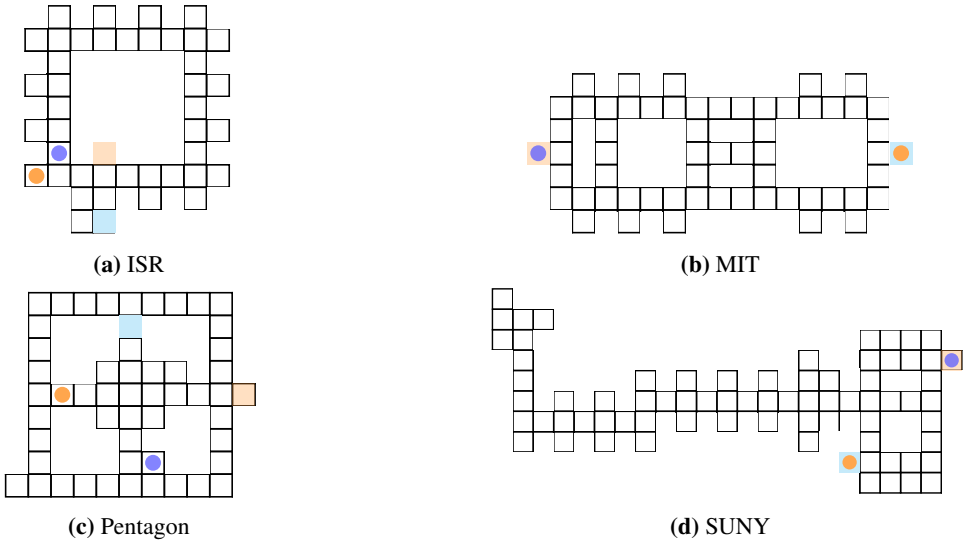
$$\texttt{Match}(p_1, p_2) = \begin{cases} 0 & \text{if } p_1 \text{ and } p_2 \text{ are equal,} \\ 1 & \text{otherwise.} \end{cases}$$

This yields the condition:

$$\texttt{Match}(p_1, p_2) < 1,$$

which expresses whether the two compared alphabets p_1 and p_2 are equal or not. The corresponding equation is:

$$1 - \texttt{Match}(p_1, p_2)$$



(a) ISR **(b)** MIT **(c)** Pentagon **(d)** SUNY

Figure 7: Maps of Grid World [35] benchmarks.

- For the symbol #, we are checking whether the alphabet under consideration is equal to #. This is expressed as the logical condition $p \leftrightarrow \#$.

- The HyperLTL formula defined for the wildfire scenario, denoted as $\varphi_{\mathsf{Rescue}}$:
  - For the properties *V1*, *V2*, $\cdots$ and *F1*, *F2*, $\cdots$, we use the same distance function `Dist` (as in SRL and DST) to calculate the distance from agents to victims and fire zones, with $c = 1$. That is, the reachability of the two objectives.
  - For the property $|\mathsf{Location}_{\tau_1} - \mathsf{Location}_{\tau_2}| < 3$, we use the distance function `Dist` with $c = 3$. This constraints make sure that for the optimized strategy, the two agents are always staying within the safe communication range.

D.2 Safe RL (SRL)

Case Study Setup. In the Safe RL case study (see Figure 7), the blue circle (denoted A1) and the orange circle (denoted A2) are agents that aim to learn optimal policies to navigate from their initial positions to their respective targets: the blue square (G1) and the orange square (G2), while avoiding collisions. The state space is represented as a tuple $\langle x, y \rangle$, and the action space is defined as $a = \{stay, up, down, left, right\}$. The following HyperLTL formula φ_{SRL} specifies the required objectives:

$$\varphi_{\text{SRL}} \triangleq \forall \tau_1 . \exists \tau_2 . \left(\Diamond \psi_{\mathsf{G1}_{\tau_1}} \wedge \Diamond \psi_{\mathsf{G2}_{\tau_2}} \wedge \Box \psi_{\mathsf{CA}_{\tau_1,\tau_2}} \right)$$

where the subformulas are defined as:

$$\psi_{\mathsf{G1}_{\tau_1}} \triangleq \langle x_{\tau_1}, y_{\tau_1} \rangle = \langle x_{G1}, y_{G1} \rangle, \qquad \psi_{\mathsf{G2}_{\tau_2}} \triangleq \langle x_{\tau_2}, y_{\tau_2} \rangle = \langle x_{G2}, y_{G2} \rangle,$$

$$\psi_{\mathsf{CA}_{\tau_1,\tau_2}} \triangleq \langle x_{\tau_1}, y_{\tau_1} \rangle \neq \langle x_{\tau_2}, y_{\tau_2} \rangle.$$

Here, $\psi_{\mathsf{G1}_{\tau_1}}$ and $\psi_{\mathsf{G2}_{\tau_2}}$ express that A1 and A2 eventually reach G1 and G2, respectively. The subformula $\psi_{\mathsf{CA}_{\tau_1,\tau_2}}$ ensures that the agents avoid collisions while navigating toward their goals.

First Experminemt Setup. In the first experiment, to ensure a fair comparison, we evaluate HYPRL combined with CQ-Learning against the shielding method [17] also using CQ-Learning. The implementation and hyperparameters of CQ-Learning are taken from [17]. Since CQ-Learning initially employs a decentralized phase (i.e., a single-agent state space), we modify the formula φ_{SRL} to remove inter-agent dependencies:

$$\forall \tau_1 . \forall \tau_2 . \left(\Diamond \psi_{\mathsf{G1}_{\tau_1}} \wedge \Diamond \psi_{\mathsf{G2}_{\tau_2}} \wedge \Box \psi_{\mathsf{CA}_{\tau_1,\tau_2}} \right)$$

https://doi.org/10.52202/085713-1743

In this setting, Skolemization is unnecessary, and the agents act independently. When the algorithm transitions into a joint phase, expanding the state space to include inter-agent interactions, we use the original specification φ_{SRL}. At this stage, HYPRL guides the learning process effectively to achieve both goal completion and collision avoidance.

Second Experminemt Setup. We employ DQN as our learning algorithm, utilizing a neural network with three hidden layers of 512 nodes and ReLU activation functions. We set the discount factor to $\gamma = 1.0$, the learning rate to 0.001, the initial ϵ to 1.0 with a decay rate of 0.995 down to a minimum of 0.01, and use the Adam optimizer. We set the number of training episodes to 200 for the SUNY, ISR, and Pentagon maps, and to 300 for the MIT map due to its increased complexity. Each episode consists of 300 steps (see Figure 7 for the map setups).

Baselines Reward Functions. The manually designed reward functions have the following form:

$$R^{\text{SRL}} = \begin{cases} a & \text{if both agents reach their respective goals,} \\ b & \text{if one agent reaches its goal,} \\ c & \text{if the agents collide.} \end{cases}$$

The function R^{SRL} addresses all objectives and safety constraints of the problem. The manually designed reward function is tested with the following values:

- $a = 10, b = 5, c = -5$ (R_1^{SRL})
- $a = 2, b = 1, c = -1$ (R_2^{SRL})
- $a = 20, b = 10, c = -10$ (R_3^{SRL})
- $a = 100, b = 50, c = -10$ (R_4^{SRL})
- $a = 10, b = 5, c = -10$ (R_5^{SRL})

In the main paper, we report the results obtained using values $a = 10, b = 5, c = -5$ (R_1^{SRL}) for the baseline reward function. Figure 8 presents the results corresponding to all different reward functions R^1, R^2, R^3, R^4, and R^5 we introduced above.

D.3 Deep Sea Treasure (DST)

Case Study Setup. This case study (see Figure 9) was originally proposed by [42]. We adopt the implementation provided in [46]. The state space is represented as a tuple $\langle x, y, step \rangle$, and the action space is defined as $a = \{up, down, left, right\}$. The following HyperLTL formula φ_{DST} specifies the required objectives:

$$\varphi_{\text{DST}} \triangleq \forall \tau_1 . \exists \tau_2 . \Diamond(T1_{\tau_1} \wedge \Diamond(T2_{\tau_1} \wedge \Diamond(T3_{\tau_1}) \dots)) \wedge \Box\,(step_{\tau_2} < \delta) \wedge \Box\,(|pos_{\tau_1} - pos_{\tau_2}| < 1)$$

The original case study is in a single-agent environment with two objectives. We modified it into a multi-agent problem by assigning one agent with the objective of maximizing collected treasure, and another agent with the task of keeping the number of steps below a certain threshold.

Experminemt Setup. For our experiments, we used PPO and DQN implementations from [38]. For DQN, we set the hyperparameters as follows: discount factor $\gamma = 0.99$, ϵ decaying from 1.0 to 0.07 over a fraction of 0.2, and a learning rate of 0.0004. Since the DQN implementation from [38] does not directly support multi-discrete actions, we expanded the action spaces of both agents and defined a unified action space compatible with this implementation. For PPO, we set hyperparameters as following, $\gamma = 0.99$, clipping factor to 0.2, learning rate to 0.0003, and GAE lambda to 0.98. Additionally, to ensure a fair comparison, we added a reward penalty of -10 to the baseline reward function whenever the two agents became separated.

D.4 Additional Experiment on DST

We conduct an additional experiment on DST, where the focus is on minimizing the step constraint in the problem. In Table 6, we report the number of steps required for the cumulative treasure achieved by the treasure collector to exceed values of 1, 5, 10, 15, 20, and 30. The results show that HYPRL consistently and significantly outperforms the baselines in all settings, except when using PPO trained for 1000 episodes.

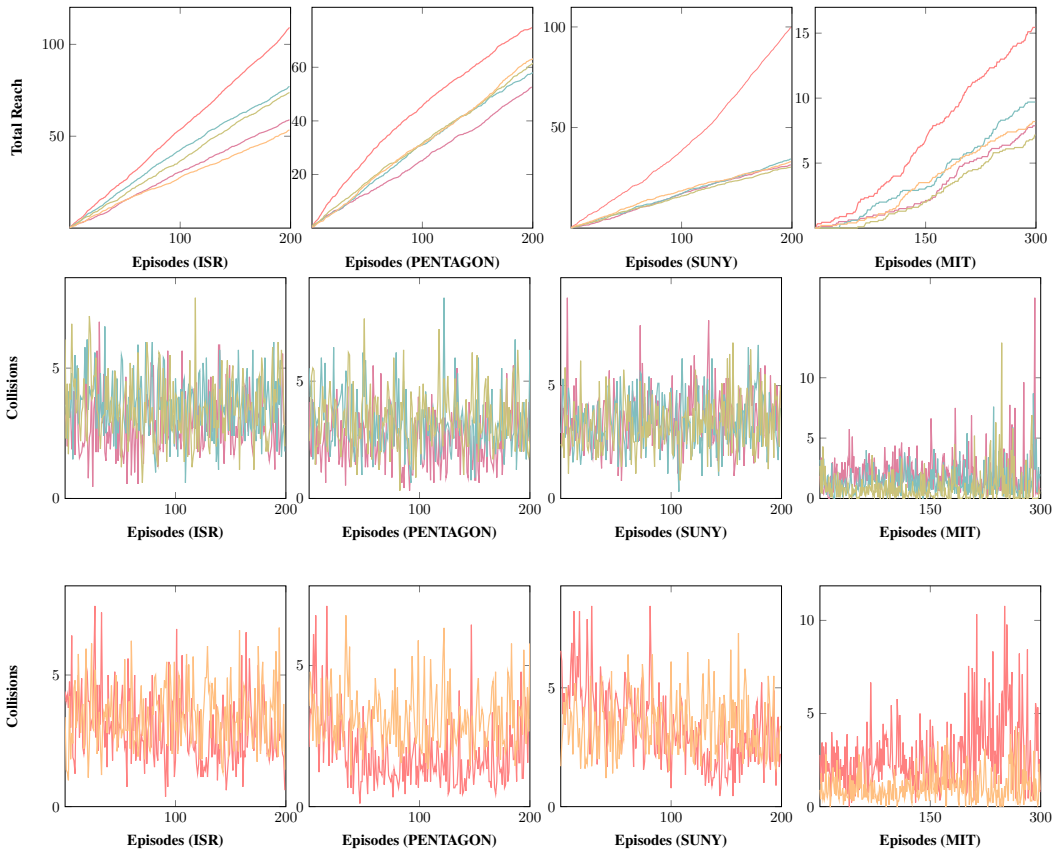


Figure 8: Total number of successful goal completions by both agents (top) and number of collisions (bottom). Lines ——, ——, ——, ——, and —— correspond to reward functions R^1, R^2, R^3, R^4, and R^5, respectively. Results are averaged over 10 runs.

D.4.1 The Post Correspondence Problem (PCP)

Case Study Setup. PCP (see Figure 11) consists of a set of k dominos, denoted as $D = \{dom_0, dom_1, \ldots, dom_k\}$. Each domino dom_i $(0 \leq i \leq k)$ is represented by a pair of nonempty finite words (top_i, bot_i) from a given alphabet Σ. The objective is to find a finite sequence of dominos such that the concatenated words on the top match those on the bottom. In our case study, the state space is defined as a tuple $\langle top, bot \rangle$ for each domino. The action space consists of selecting a domino from D, resulting in k possible actions $A = \{dom_0, dom_1, \ldots, dom_k\}$. Notably, traces (i.e., agents) are unaware of the context of the dominos and can only identify them by their labels (i.e., dom_i). The initial state of the MDP encodes empty words on both the top and bottom: $s^0 = \langle \varepsilon, \varepsilon \rangle$.

Table 6: DST results on steps required to achieve the desired treasures, showing average and standard error over 10 evaluations.

Method	Episodes	Treasures Achieved					
		1	5	10	15	20	30
PPO	500	199.01±6.65	218.66±6.67	244.87±6.68	247.89±6.67	253.16±6.66	268.01±6.70
PPO +HYPRL	500	**13.30**±1.01	**29.72**±1.57	**39.10**±1.81	**49.22**±1.96	**56.59**±2.03	**71.55**±2.26
PPO	1000	**5.75**±0.30	**6.75**±0.30	**29.68**±0.76	**32.29**±0.75	**36.09**±0.74	**39.82**±0.72
PPO +HYPRL	1000	55.37±3.66	106.89±4.96	126.25±5.10	141.06±5.27	153.49±5.35	183.51±5.70
DQN	500	832.10±9.93	897.54±6.95	916.40±5.63	917.64±5.55	917.89±5.54	926.33±5.25
DQN +HYPRL	500	**114.60**±2.67	**115.61**±2.56	**116.61**±2.52	**117.61**±2.50	**148.66**±4.34	**155.39**±4.58
DQN	1000	893.37±9.67	953.25±6.55	992.10±2.50	992.12±2.55	996.05±1.52	997.19±1.14
DQN +HYPRL	1000	**303.63**±6.64	**319.09**±6.38	**335.20**±6.21	**335.20**±6.21	**337.21**±6.18	**339.90**±6.22

https://doi.org/10.52202/085713-1743

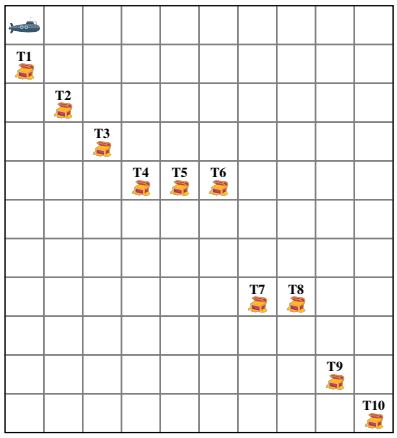


Figure 9: DST Grid Map.

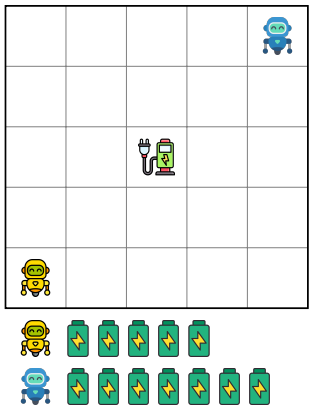


Figure 10: Job Scheduling benchmark.

Subsequently, they choose actions from the action space (choosing a domino) to construct traces that aims to satisfy the PCP objective.

Before encoding PCP in HyperLTL, we note that in [19, 9], the authors reduce PCP to the satisfiability problem for the $\forall\exists$ fragment of HyperLTL and the emptiness problem of nondeterministic finite-state hyper automata. Part of the encoding is to ensure that only valid dominos are chosen. We do not need those constraints in our encoding, as the action space of the MDP enforces choosing valid dominos only. This is the reason our that encoding is less complex than that of [19, 9].

We formalize a valid solution to PCP in HyperLTL as follows. The set AP of atomic propositions is defined such that as $\Sigma = 2^{\mathsf{AP}} \cup \{\#\}$, where $\#$ encodes termination. Essentially, the HyperLTL formula requires that for all traces τ_1, where top and bottom words match up to the end of the shorter trace, there exists a trace τ_2 such that τ_1 is a SemiMatch and τ_2 extends τ_1 to complete equal top and bottom words:

$$\varphi_{\mathsf{PCP}} \triangleq \forall\tau_1.\exists\tau_2.\ \psi_{\mathsf{SemiMatch}_{\tau_1}}\ \mathcal{U}\ \left(\psi_{\mathsf{Extend}_{\tau_1,\tau_2}} \wedge \psi_{\mathsf{Match}_{\tau_2}}\right)$$

where $\varphi_{\mathsf{SemiMatch}}$ means the top and bottom words match up to the length of the shorter word:

$$\psi_{\mathsf{SemiMatch}_{\tau_1}} \triangleq \left[\bigwedge_{p\in\mathsf{AP}} (p_{top_{\tau_1}} \leftrightarrow p_{bot_{\tau_1}})\right]\ \mathcal{U}\ (\#_{top_{\tau_1}} \oplus \#_{bot_{\tau_1}})$$

where '$\oplus$' is the xor operator. The formula φ_{Match} indicates that the word constructed on the top and bottom are equal:

$$\psi_{\mathsf{Match}_{\tau_2}} \triangleq \square \bigwedge_{p\in\mathsf{AP}} (p_{top_{\tau_2}} \leftrightarrow p_{bot_{\tau_2}})$$

Finally, formula $\varphi_{\mathsf{Extend}_{\tau_1,\tau_2}}$ encodes that trace τ_2 is a successor trace τ_1 as follows:

$$\varphi_{\mathsf{Extend}_{\tau_1,\tau_2}} \triangleq \left[\bigwedge_{p\in\mathsf{AP}} \left((p_{top_{\tau_1}} \leftrightarrow p_{top_{\tau_2}}) \wedge (p_{bot_{\tau_1}} \leftrightarrow p_{bot_{\tau_2}})\right)\right] \mathcal{U}$$
$$\left((\#_{top_{\tau_1}} \vee \#_{bot_{\tau_1}}) \wedge (\neg\#_{top_{\tau_2}} \wedge \neg\#_{bot_{\tau_2}})\right)$$

Experminemt Setup. We employ DQN as our learning algorithm, utilizing a neural network with three layers of 512 nodes and ReLU activation functions. We set the discount factor γ to 0.99, the learning rate to 0.001, and the initial ϵ to 1.0 with a decay rate of 0.995 down to a minimum of 0.001. We use the Adam optimizer and train for 1000 episodes. This experiment is conducted for domino collections containing 5 and 6 dominos.

Baselines Reward Functions. The manually designed reward functions have the following form:

$$R^{\mathrm{PCP}} = \begin{cases} a & \text{if the same indexed letters on the top and bottom are equal,} \\ b & \text{otherwise.} \end{cases}$$

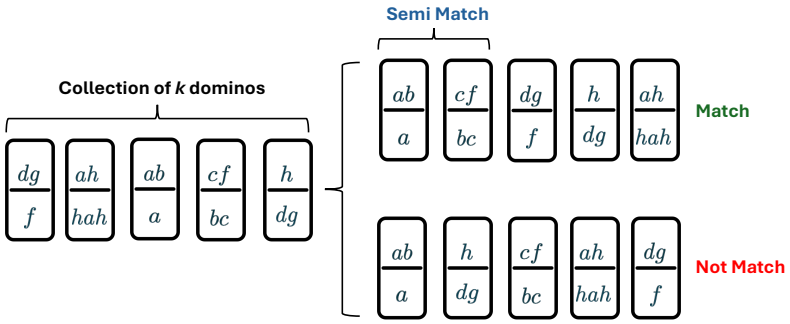


Figure 11: PCP overview

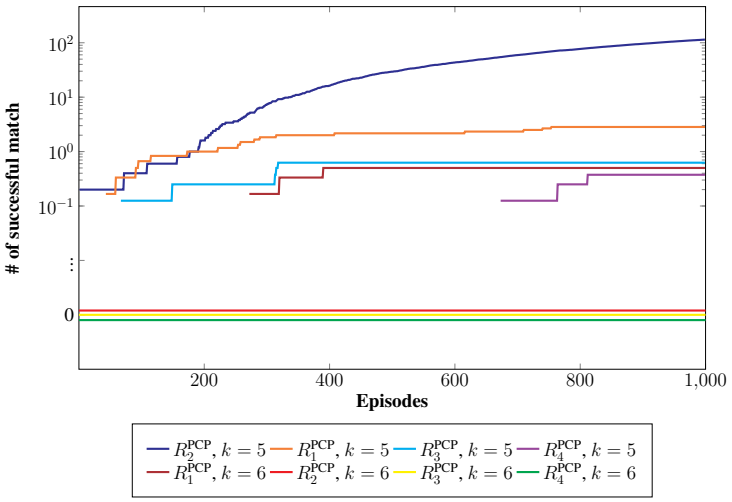


Figure 12: PCP baseline results showing average total matches over 10 runs.

The function R^{PCP} addresses all objectives and safety constraints of the problem. The manually designed reward function is tested with the following values:

- $a = 1, b = -1$ (R_1^{PCP})
- $a = 5, b = -2$ (R_2^{PCP})
- $a = 10, b = -10$ (R_3^{PCP})
- $a = 1, b = -5$ (R_4^{PCP})

In the paper, we report results using the baseline reward function with values $a = 5$, $b = -2$ (R_2^{PCP}) for $k = 5$, and $a = 1$, $b = -1$ (R_1^{PCP}) for $k = 6$. Figure 12 presents the results corresponding to the different reward functions. The baselines R_2^{PCP}, R_3^{PCP}, and R_4^{PCP} for $k = 6$ did not successfully produce any matches. We visualize this using a value of zero in the log-scale plot shown in Figure 12.

Table 7: Wildfire results, avg. and std. error of 10 trials after 5k episodes.

Size	Method	Dist	Steps O_1	Steps O_2
3^2	PPO + R_1^{FAIR}	2.8±0.01	77.13±5.7	T/O
	PPO + R_2^{FAIR}	**2.5**±0.01	**33.43**±4.1	**787.03**±31.8
5^2	PPO + R_1^{FAIR}	5.6±0.02	**31.2**±4.5	**7057.8**±1047.9
	PPO + R_2^{FAIR}	**4.2**±0.01	62.7±9.7	T/O

https://doi.org/10.52202/085713-1743

D.4.2 Wild Fire Scenario

Case Study Setup. Consider a wildfire scenario in a grid-world environment as we introduced in Section 1. Three locations are on fire, and two victims need to be rescued. Two autonomous agents are deployed from the same position with distinct objectives: the firefighter agent (FF) must extinguish all fire zones, and the medical agent (Med) must rescue all victims. The agents must also satisfy two constraints: which requires them to always remain within a 2-cell communication range; and which prohibits Med from entering any fire zone before FF has extinguished the fire in that zone. The state space is represented as a tuple $\langle x, y \rangle$, and the action space is defined as $a = \{\textit{stay}, \textit{up}, \textit{down}, \textit{left}, \textit{right}\}$. The goal is to learn optimal policies for FF and Med that maximize the probability of satisfying all these requirements. The following HyperLTL formula captures the objectives of this case study:

$$\varphi_{\mathsf{Rescue}} \triangleq \forall\tau_1.\exists\tau_2.(\psi_{\mathsf{fire}} \wedge \psi_{\mathsf{save}} \wedge \psi_{\mathsf{dist}} \wedge \psi_{\mathsf{safe}})$$

$$O_1 :\ \psi_{\mathsf{fire}} \triangleq \Diamond(i_{\tau_1}) \wedge \Diamond(f_{\tau_1}) \wedge \Diamond(c_{\tau_1}) \qquad C_1 :\ \psi_{\mathsf{dist}} \triangleq \Box(|\mathsf{Location}_{\tau_1} - \mathsf{Location}_{\tau_2}| < 3)$$

$$O_2 :\ \psi_{\mathsf{save}} \triangleq \Diamond(g_{\tau_2}) \wedge \Diamond(f_{\tau_2}) \qquad C_2 :\ \psi_{\mathsf{safe}} \triangleq (\neg i_{\tau_2}\, \mathcal{U}\, i_{\tau_1}) \wedge (\neg f_{\tau_2}\, \mathcal{U}\, f_{\tau_1}) \wedge (\neg c_{\tau_2}\, \mathcal{U}\, c_{\tau_1})$$

Experiment Setup. We use the PPO implementation from [38], with the following hyperparameters: a learning rate of 0.0003, a clipping parameter of 0.2, a discount factor $\gamma = 0.995$, and a GAE lambda of 0.95. We conduct our experiments on 3×3 and 5×5 grid-world environments. The timeout is set to 1000 steps for the 3×3 grid and 10,000 steps for the 5×5 grid.

Baselines. We evaluated (see Table 7) two baseline reward functions in this experiment:

- R_1^{FAIR}: extinguishing fire: $+50$, rescuing a victim: $+10$, agent out of range: -100, and Med in a fire zone: -100.
- R_2^{FAIR}: Extinguishing fire: $+10$, rescuing a victim: $+50$, agent out of range: -100, and Med in a fire zone: -100.

We used R_2^{FAIR} in the paper, although in 5×5 R_1^{FAIR} performs better than R_2^{FAIR} (in terms of only Steps O_1 and Steps O_2). However, we chose to include R_2^{FAIR} in the main paper because its performance in the 3×3 grid is consistently better than R_1^{FAIR} across all metrics. This trend does not hold in the 5×5 grid, where R_1^{FAIR} outperforms R_2^{FAIR} only in Steps O_1 and Steps O_2.

D.5 Additional Experiment

Case Study Setup. In this case study (see Figure 10), a single permanent resource is placed on a grid with multiple agents, which must learn to share resources. The objective here is to maximize the overall utility of all agents while ensuring fair allocation of the resource. In other words, the goal is not merely to maximize utility by allocating the resource to a single agent but to distribute it equitably among all agents. The action space is defined as $a = \{\textit{stay}, \textit{up}, \textit{down}, \textit{left}, \textit{right}\}$. The state space is extended by $\langle x, y, \mathsf{Energy} \rangle$. We expresses our allocation and fairness objectives by the following HyperLTL formula:

$$\varphi_{\text{FAIR}} \triangleq \forall\tau_1.\forall\tau_2.\ \Big(\Box\Diamond \mathsf{Resource}_{\tau_1} \ \wedge\ \Box\Diamond \mathsf{Resource}_{\tau_2}\Big) \wedge \Box\big(|\mathsf{Energy}_{\tau_1} - \mathsf{Energy}_{\tau_2}| < \delta\big)$$

That is, all agents should eventually gain access to the resource at every step, while ensuring that the difference in their Energy levels (i.e., allocated resources) remains less than a threshold δ, which can be set as a hyperparameter[2]. In our setup, agents start with $\mathsf{Energy} = 0$, and each time an agent reaches the resource position, its energy level increments by one while maintaining the same action space.

Experiment Setup. We employ PPO as our learning algorithm, using two neural networks: a policy network and a value network. Both networks consist of three hidden layers with 512 nodes and ReLU activation functions. The policy network uses a softmax activation in the output layer, while the value network uses a linear output. We set the learning rate to 0.001, the discount factor $\gamma = 0.95$, and the clipping factor to 0.2. Optimization is performed using the Adam optimizer. We also set $\delta = 10$.

[2]We acknowledge that $\Box\Diamond$ is a Büchi condition and is strange to be use in finite semantics. Nevertheless, when it is interpreted in the context of robustness values, function ρ attempts to maximize the occurrence of Resource, which is the intended objective.

Baselines Reward Functions. The manually designed reward functions have the following form:

$$R^{\text{FAIR}} = \begin{cases} a & \text{if resource allocated to either of agents,} \\ b & \text{if } |\mathsf{Energy}_{\text{Agent 1}} - \mathsf{Energy}_{\text{Agent 2}}| > \delta \end{cases}$$

The function R^{FAIR} addresses all objectives and safety constraints of the problem. The manually designed reward function is tested with the following values:

- $a = 2, b = -1$ (R_1^{FAIR})
- $a = 20, b = -5$ (R_2^{FAIR})
- $a = 10, b = -5$ (R_3^{FAIR})
- $a = 5, b = -5$ (R_4^{FAIR})

Analysis and Results. The evaluation (see Figure 13) demonstrates how the learning process maximizes utility for both agents while minimizing the difference in their utilities. In Figure 13a, using HYPRL, we observe that the two agents initially start with low resource utilization. Although this issue is addressed over time, there is a noticeable gap in resource allocation between episodes 150 and 270. HYPRL gradually reduces this disparity, and after episode 400, it achieves high resource utilization while maintaining fairness. This is evident from the fact that the gap between the maximum and minimum utilization becomes nearly invisible between episodes 400 and 500. By the end of training, each agent receives approximately 40 to 45 resources per episode (with 100 steps), while the optimal allocation is 50 resources per agent. These results highlight the effectiveness of HYPRL in achieving fairness in MARL.

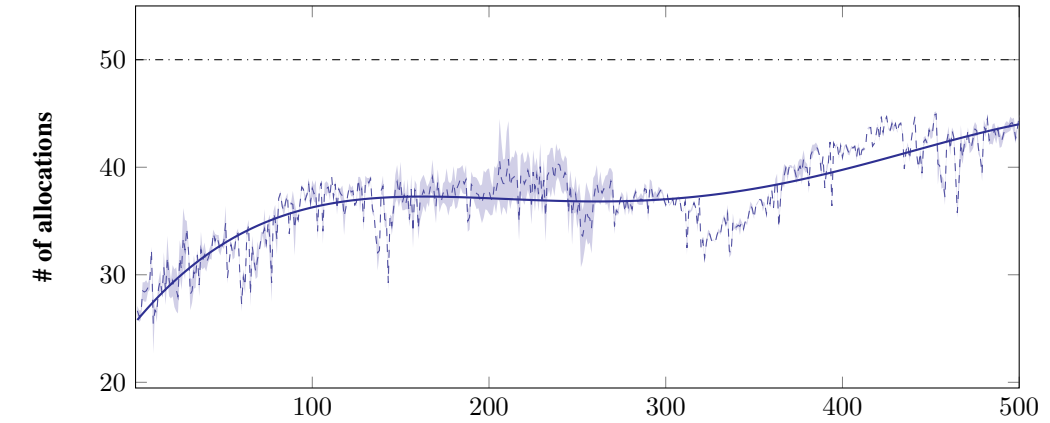
On the other hand, using PPO with the baseline reward functions does not successfully yield a policy that both maximizes the average utility of the agents and minimizes the gap between the minimum and maximum allocated resources. For example, R_1^{FAIR} (see Figure 13b) succeeds in maximizing the average resource allocation to the optimal level but fails to minimize the disparity between agents in terms of resource allocation. The other baseline reward functions R_2^{FAIR}, R_3^{FAIR}, and R_4^{FAIR} exhibit similar behavior (see Figures 13c to 13e, respectively). While they perform better than R_1^{FAIR} in minimizing the min-max range of resource allocation, they still fall short compared to HYPRL in achieving both fairness and optimal utility.

E Why HYPRL is Important?

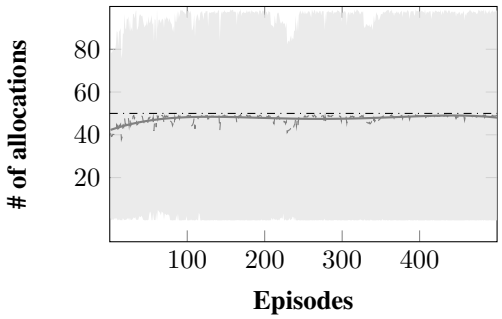
In this work, we propose a method named HYPRL, which introduces a novel fusion between reinforcement learning and a recently developed class of logics known as hyperproperties. Hyperproperties have gained increasing attention in the formal methods community due to their expressive power in capturing system-level behaviors that span across multiple execution traces. We observe that many objectives and constraints in MARL naturally align with the expressive capabilities of hyperproperties. Motivated by this insight, we present a framework that first formally specifies the objectives and constraints of a given RL problem using hyperproperties and then learns policies that aim to maximize the probability of satisfying these specifications.

We believe that our approach provides a solid theoretical foundation for a new direction in reinforcement learning, where users can formally express their high-level requirements using hyperproperties and then synthesize policies that satisfy them. Importantly, hyperproperties are not limited to specifying objectives of individual agents or pairwise dependencies. Recent extensions such as HyperATL [7] allow reasoning about team-level objectives and inter-team relational dependencies, further expanding the scope of formal specification in MARL.

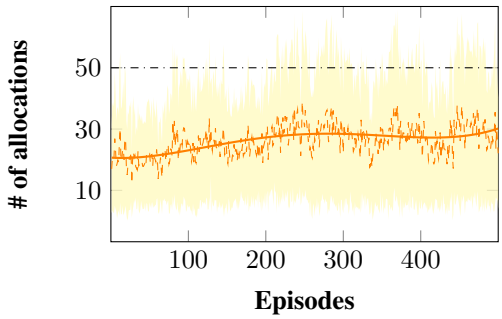
https://doi.org/10.52202/085713-1743



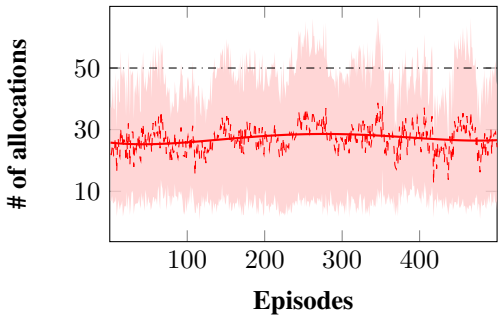
(a) HYPRL



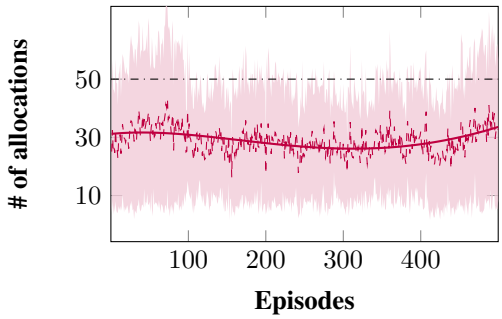
(b) R_1^{FAIR} baseline reward function.



(c) R_2^{FAIR} baseline reward function.



(d) R_3^{FAIR}baseline reward function.



(e) R_4^{FAIR} baseline reward function.

Figure 13: Fair resource allocation results under various baseline reward functions. Boundaries represent the minimum and maximum allocated resources per episode. Dashed lines indicate average usage, dashed-dotted lines show optimal allocations, and solid lines represent polynomial trend fits. Results are based on 10 independent runs in a 4×4 grid with two agents, each run consisting of 500 episodes with 100 steps per episode.