Mining Causal Signal Temporal Logic Formulas for Efficient Reinforcement Learning with Temporally Extended Tasks

Hadi Partovi Aria HPARTOVI@ASU.EDU and **Zhe Xu*** XZHE1@ASU.EDU
School for Engineering of Matter, Transport and Energy, Arizona State University, Tempe, AZ, USA
**Corresponding author*

Editors: G. Pappas, P. Ravikumar, S. A. Seshia

Abstract

Reinforcement Learning (RL) has emerged as a powerful paradigm for solving sequential decision-making problems. However, traditional RL methods often lack an understanding of the causal mechanisms that govern the dynamics of an environment. This limitation results in inefficiencies, challenges in generalization, and reduced interpretability. To address these challenges, we propose Signal Temporal Logic Causal Inference RL (STL-CIRL), a framework that mines interpretable causal specifications through Signal Temporal Logic and reinforcement learning, using counterexample-guided refinement to jointly optimize policies and causal formulas. We compare the performance of agents leveraging explicit causal knowledge with those relying solely on traditional RL approaches. Our results demonstrate the potential of causal reasoning to enhance the efficiency and robustness of RL for complex tasks.

Keywords: Reinforcement Learning, Causal Inference, Signal Temporal Logic

1. Introduction

Reinforcement Learning (RL) has become a cornerstone of artificial intelligence, solving problems from robotic control to healthcare. Despite its achievements, conventional RL techniques typically function as opaque mechanisms that fail to capture the underlying causal relationships governing environmental dynamics. This limitation leads to inefficient learning, poor generalization, and reduced interpretability. Addressing these challenges requires incorporating causal reasoning into RL. Causal inference provides a systematic way to understand how variables influence one another, enabling agents to predict outcomes and explain decisions. However, existing RL frameworks seldom integrate causal reasoning, relying instead on exhaustive exploration (Bareinboim (2020)). Causal Signal Temporal Logic (Causal STL) bridges this gap by formalizing causal relationships with temporal properties, enabling the specification of cause-effect relationships critical for RL tasks with complex temporal dependencies (Deng et al. (2023)).

Related Works: Recent research has integrated causal reasoning and temporal logic into RL to enhance efficiency and interpretability. Dasgupta et al. (2019) demonstrated causal reasoning through meta-RL, showing agents can make causal inferences in novel situations. Li et al. (2017) used temporal logic to specify complex tasks, while Ding et al. (2023) augmented goal-conditioned RL with causal graphs for improved generalization. This paper proposes integrating Causal STL-derived formulas into RL to evaluate how causal knowledge affects sample efficiency, and robustness.

Contributions: This paper makes three primary contributions. First, we propose a framework using RL to extract causal temporal logic formulas for more efficient exploration of relevant state-action pairs. Second, we provide theoretical guarantees for convergence to optimal causal formulas and

policies, with established sample complexity bounds. Third, we introduce dynamic counterfactual traces using Gaussian Process models to simulate alternative scenarios, enabling agents to jointly discover and exploit causal knowledge for improved learning efficiency. These traces simulate how different causes might change outcomes, revealing hidden cause-effect relationships.

1.1. Motivation: Gene Modification for Disease Treatment

Gene modification strategies in medical applications involve regulatory networks with causal dependencies, where altering gene *A* may require first activating gene *B*, while gene *C* must remain unchanged to avoid complications. A naive RL approach would inefficiently test all possible sequences, risking costly failed interventions. In contrast, RL agents with causal knowledge focus exploration on biologically valid pathways, reducing the search space and enabling faster convergence to safe treatment protocols. This example demonstrates how causal modeling improves learning efficiency, generalization, and decision interpretability in complex tasks.

2. Preliminaries

2.1. Syntax of Causal STL

The syntax of Causal STL builds upon STL, enabling the formalization of causal relationships. A typical Causal STL formula is expressed as (Deng et al. (2023)):

$$\Phi ::= \text{do}(\phi_c) \rightsquigarrow \phi_e, \tag{1}$$

where ϕ_c represents the cause formula, and ϕ_e represents the effect formula. These formulas use STL operators: $\Diamond_{[a,b]}\phi$ (eventually) indicates ϕ holds at some point within $[a, b]$; $\Box_{[a,b]}\phi$ (always) means ϕ holds throughout $[a, b]$; and $X(t) \sim d$ represents a condition where variable X at time t satisfies relation $\sim \in \{\leq, <, \geq, >\}$ compared to threshold $d \in \mathbb{R}$.

2.2. Qualitative Semantics of Causal STL

The qualitative semantics of Causal STL define when a formula is satisfied within a given system. A Causal STL formula $\Phi := \text{do}(\phi_c) \rightsquigarrow \phi_e$ is satisfied if (Deng et al. (2023)): **Sufficiency:** For all interventions $\text{do}(\phi_c)$, the effect formula ϕ_e holds: $\forall \text{do}(\phi_c),\ \phi_e$ holds; **Necessity:** If the effect ϕ_e holds, then the cause ϕ_c must have been intervened upon: $\phi_e \implies \text{do}(\phi_c)$.

2.3. Quantitative Semantics of Causal STL

To quantify the strength of causal relationships, Causal STL introduces metrics for sufficiency and necessity based on empirical data. The sufficiency degree is defined as:

$$S(\theta; \mathcal{D}) = \frac{1}{|\mathcal{D}_+|} \sum_{\tau \in \mathcal{D}_+} \rho(\tau, \phi_e, t \mid \text{do}(\phi_c)), \tag{2}$$

where $\mathcal{D}_+$ is the subset of trajectories in dataset $\mathcal{D}$ where $\rho(\tau, \phi_c, 0) > 0$. The necessity degree is formulated as:

$$N(\theta; \mathcal{D}) = -\frac{1}{|\mathcal{D}_-|} \sum_{\tau \in \mathcal{D}_-} \rho(\tau, \phi_e, t \mid \text{do}(\neg\phi_c)), \tag{3}$$

where $\mathcal{D}_-$ is the subset of trajectories in dataset $\mathcal{D}$ where $\rho(\tau, \phi_c, 0) < 0$. Here, $\rho(\tau, \phi, t)$ represents the robustness degree of trajectory τ with respect to formula ϕ at time t, a quantitative measure of

how strongly τ satisfies or violates ϕ. Positive values indicate satisfaction, with higher values representing stronger satisfaction, while negative values indicate violation. The formal definition and calculation of ρ for different STL operators are provided in Appendix 9.2.

2.4. Inference of Causal STL Formulas

The inference of Causal STL formulas involves identifying cause-effect relationships that explain observed behaviors within a dataset $\mathcal{D}$. Let θ denote the parameters that define a candidate cause formula $\phi_c(\theta)$, such as involved variables, temporal bounds, and thresholds. The process begins with an initial cause formula, ϕ_c^0, which can be derived from domain knowledge, a predefined set of common causal patterns or templates (e.g., "variable X exceeds threshold d_X within interval $[t_1, t_2]$"). This process is formulated as an optimization problem, where the objective function maximizes the explanatory power of a Causal STL formula Φ_i, defined as:

$$\sup_{\theta \in \Theta_c} J(\theta; \mathcal{D}) = -E(\theta; \mathcal{D}) + \lambda_S S(\theta; \mathcal{D}) + \lambda_N N(\theta; \mathcal{D}), \tag{4}$$

where λ_S and λ_N are positive parameters that balance the degrees of existence, sufficiency and necessity, and $E(\theta; \mathcal{D})$ quantifies the degree of existence in the dataset and is given by:

$$E(\theta; \mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{\tau \in \mathcal{D}} e^{-(\rho(\tau, \phi_c(\theta), t) - \rho(\tau^*, \phi_c(\theta), t))}, \tag{5}$$

with $\rho(\tau, \phi_c(\theta))$ representing the robustness degree of trajectory τ with respect to the parameterized cause formula $\phi_c(\theta)$. Also, τ^* is a reference trajectory serving as a baseline, typically chosen to represent nominal or desired system behavior. The difference in robustness between τ and τ^* normalizes the existence measure, so trajectories similar to the reference contribute less, indicating the cause formula captures common rather than rare patterns.

3. Q-Learning with Reward Functions for STL Objectives

Q-learning is a model-free reinforcement learning algorithm that learns state-action values in MDPs (Corazza et al. (2024)). Standard Q-learning must be adapted for Signal Temporal Logic objectives by incorporating custom reward functions that align with STL satisfaction measures (Aksaray et al. (2016)). Standard Q-learning optimizes an action-value function $Q(s, a)$ that evaluates state-action pairs. The update rule is (Alsadat et al. (2024)):

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right], \tag{6}$$

where α is the learning rate, $\gamma \in (0, 1)$ is the discount factor, and r is the immediate reward for taking action a in state s, and s' is the next state. STL objectives pose challenges as satisfaction depends on entire trajectories, not individual states, and rewards can only be computed after observing complete trajectories (Venkataraman et al. (2020)).

3.1. Reward Function for STL

To adapt Q-learning for Signal Temporal Logic (STL) objectives, which depend on entire trajectories rather than individual state transitions, we employ a custom reward function based on the

robustness degree of STL formulas, as introduced in Aksaray et al. (2016). To handle the history dependence inherent in STL specifications, the system is modeled as a τ-MDP, where each state $s^\tau \in \Sigma^\tau$(state space) represents a sequence of τ consecutive states from the original MDP, with τ determined by the horizon of the inner formula ϕ. For an STL specification Φ, the robustness degree quantifies how well a trajectory satisfies the formula. Specifically, for a trajectory $\tau = (s_0, s_1, \ldots, s_T)$, the robustness is defined differently depending on the temporal operator. If $\Phi = \Diamond_{[0,T]}\phi$ (eventually), the robustness is defined as $\rho(\tau, \Diamond_{[0,T]}\phi, 0) = \max_{t\in[0,T]} \rho(\tau, \phi, t)$, meaning ϕ must hold at least once within $[0, T]$. If $\Phi = \Box_{[0,T]}\phi$ (always), the robustness is defined as $\rho(\tau, \Box_{[0,T]}\phi, 0) = \min_{t\in[0,T]} \rho(\tau, \phi, t)$, requiring ϕ to hold continuously over $[0, T]$. Here, $\rho(\tau, \phi, t)$ measures the robustness of trajectory τ with respect to the inner formula ϕ at time t. When time is unspecified, $\rho(\tau, \phi)$ defaults to evaluation at $t = 0$. In the τ-MDP framework, the reward function is defined based on the robustness degree of the current τ-state s^τ. For a transition from s_t^τ to s_{t+1}^τ via action a, the reward $R(s_{t+1}^\tau)$ is:

$$R(s_{t+1}^\tau) = \begin{cases} e^{\beta\rho(s_{t+1}^\tau, \phi)}, & \text{if } \Phi = \Diamond_{[0,T]}\phi, \\ -e^{-\beta\rho(s_{t+1}^\tau, \phi)}, & \text{if } \Phi = \Box_{[0,T]}\phi, \end{cases} \tag{7}$$

where $\beta > 0$ is a scaling parameter. For $\Phi = \Diamond_{[0,T]}\phi$, the exponential reward encourages maximizing robustness at some point in the trajectory. For $\Phi = \Box_{[0,T]}\phi$, the negative exponential penalizes low robustness, promoting continuous satisfaction. The Q-learning algorithm is then tailored to this τ-MDP setting, optimizing the action-value function $Q(s^\tau, a)$ with the update rule:

$$Q(s_t^\tau, a) \leftarrow Q(s_t^\tau, a) + \alpha \left[R(s_{t+1}^\tau) + \gamma \max_{a'} Q(s_{t+1}^\tau, a') - Q(s_t^\tau, a) \right], \tag{8}$$

where $R(s_{t+1}^\tau)$ is the robustness-based reward at the next τ-state s_{t+1}^τ. The policy $\pi(s^\tau)$ is derived by selecting the action that maximizes $Q(s^\tau, a)$, with ϵ-greedy exploration to balance exploration and exploitation (Aksaray et al. (2016)). This approach ensures that Q-learning effectively optimizes STL objectives by leveraging trajectory history and robustness-based rewards.

4. Coupled RL with Bayesian Optimization for Cause-and-Effect Satisfaction

This section presents STL-CIRL, our framework that jointly optimizes reinforcement learning policies and causal STL formulas. Algorithm 2 outlines the main process: the agent explores the environment to update its policy via Q-learning while collecting counterexample trajectories. Algorithm 1 analyzes these trajectories to compute sufficiency (S), necessity (N), and existence (E) scores, quantifying how well a candidate cause formula explains observed effects. These metrics guide Bayesian optimization to refine the causal formula, creating a feedback loop where policy learning and causal discovery mutually enhance each other.

The proposed approach integrates RL with Bayesian optimization in a closed-loop framework, where trajectory data informs the refinement of causal formulas. Each trajectory τ consists of tuples spanning a temporal horizon T: $\tau = \{(s_t, a_t, r_t, s_{t+1})\}_{t=0}^T$. These trajectories capture both the sequential nature of the learning process and the temporal evolution of cause-effect relationships. The learning process begins by initializing the RL environment, Q-values $Q(s^\tau, a)$, policy $\pi(s^\tau)$, and a Gaussian Process model for Bayesian optimization (Algorithm 2). A candidate cause formula ϕ_c^0 provides the initial structure for causal reasoning. During each episode, the agent explores the environment while maintaining a trajectory buffer τ_{cur} that tracks state-action sequences.

Algorithm 1 Evaluate Sufficiency, Necessity, and Existence

1: Initialize empty lists `sufficiency_scores`, `necessity_scores`, `existence_scores`
2: **for** $i = 1$ to I **do**
3: Get $\tau \in \mathcal{CE}$
4: Generate counterfactual τ' under $do(\pi_c')$
5: Compute $\rho(\tau', \phi_c, t)$ and $\rho(\tau', \phi_e, t)$
6: **if** $\rho(\tau', \phi_c) > \epsilon_{d_1}$ **then**
7: Append $\rho(\tau', \phi_e)$ to `sufficiency_scores`
8: **end if**
9: **if** $\rho(\tau', \phi_c) < -\epsilon_{d_2}$ **then**
10: Append $\rho(\tau', \phi_e)$ to `necessity_scores`
11: **end if**
12: Append $\rho(\tau', \phi_c)$ to `existence_scores`
13: **end for**
14: $S \leftarrow \text{Mean}(\texttt{sufficiency\_scores})$
15: $N \leftarrow e^{-(\text{Mean}(\texttt{necessity\_scores}))}$
16: $E \leftarrow e^{-(\text{Mean}(\texttt{existence\_scores}))}$
17: **return** (S, N, E)

The system continuously evaluates trajectory robustness $\rho(\tau, \phi_c, t)$ and $\rho(\tau, \phi_e, t)$ for both cause and effect formulas. When a trajectory violates the effect formula ($\rho(\tau, \phi_e, t) \leq 0$), it is stored as a counterexample in buffer $\mathcal{CE}$ (Algorithm 2, line 12). These counterexamples are crucial for computing the sufficiency, necessity, and existence measures that guide formula refinement. For each counterexample, the system generates counterfactual traces τ' by modifying state variables according to intervention rules $do(\pi_c')$ (Algorithm 1, lines 3-4). The formula refinement process optimizes the objective function $J(\phi_c) = -E + \lambda_S S + \lambda_N N$ where S, N, and E represent sufficiency, necessity, and existence measures respectively. These measures are computed by analyzing the robustness values of both original and counterfactual trajectories across different thresholds ϵ_{d_1} and ϵ_{d_2} (Algorithm 1, lines 5-12). Bayesian optimization guides the search for improved cause formulas by maintaining a probabilistic model of the objective function $J(\phi_c)$. The GP model uses a radial basis function kernel, defined as (Seeger (2004)):

$$k(x, x') = \exp\left(-\frac{1}{2l^2}||x - x'||^2\right) \tag{9}$$

where $l > 0$ is the length scale parameter that determines the smoothness of the function and how quickly the correlation between points decays with distance. This kernel enables the model to interpolate between observed formula performances and suggest promising candidates through Upper Confidence Bound (UCB) acquisition. In Algorithm 2, the cause formula ϕ_c is implemented as a parameterized temporal logic template (e.g., $\square_{[t_1,t_2]}(X \sim d)$) with the Gaussian Process optimizing parameter vector $\theta = (t_1, t_2, d)$. While the effect formula ϕ_e is provided externally as part of the task specification, our approach focuses on refining ϕ_c by maximizing the causal sufficiency and necessity scores through iterative optimization.

5. Counterexample Generation Method

Our framework employs a systematic method for generating and analyzing counterexamples to learn causal relationships effectively. We implement an iterative refinement process that combines

Algorithm 2 STL-CIRL

1: Initialize $Q(s^\tau, a) \leftarrow 0$, policy π, GP model, $\mathcal{C} \leftarrow \emptyset$
2: Set $\phi_c \leftarrow \phi_c^0$
3: **for** $k = 1$ to K **do**
4: Reset $\mathcal{E}$, get s_0, initialize $\tau_{\text{cur}} \leftarrow \emptyset$
5: **for** $t = 0$ to $T - 1$ **do**
6: Select $a_t \sim \pi(s_t^\tau)$ (ϵ-greedy)
7: Execute a_t, observe s_{t+1}, update τ_{cur}
8: Compute $\rho(\tau_{\text{cur}}, \phi_c, 0)$, $\rho(\tau_{\text{cur}}, \phi_e, 0)$
9: Compute reward
10: Update $Q(s_t^\tau, a_t)$ and $\pi(s_t^\tau)$
11: **if** $\rho(\phi_e, \tau) \leq 0$ **then**
12: Add τ_{cur} to $\mathcal{CE}$
13: **end if**
14: **end for**
15: Compute S, N, E using Algorithm 1
16: $\phi_c^{k+1} \leftarrow \arg\max_{\phi_c}(-E + \lambda_S S + \lambda_N N)$
17: Update GP model with S, N, E
18: **end for**
19: **return** (ϕ_c, π^*)

state perturbation analysis with counterexample-guided synthesis. First, it explores the state space through targeted perturbations of state variables, scaled appropriately to maintain physical feasibility. The process begins by initializing an empty set of counterexamples (line 1) and obtaining the current trajectory (line 2). When a violation of the effect formula is detected (line 3), the algorithm systematically explores perturbations of each state variable (line 4). For each variable, both positive and negative perturbations are tested within a specified range ϵ (line 5). Second, it leverages discovered counterexamples to simultaneously improve both the system specification and the control policy. Each perturbed state is generated using the *PerturbState* function (line 6), which modifies a specific state variable v_i by a perturbation value δ (line 5) to create a new state s'_t. This is followed by trajectory simulation from this new state (line 7). Valid counterexamples that reveal meaningful violations of the specifications are identified (line 8) and added to the collection (line 9). Finally, the complete set of discovered counterexamples is returned (line 14) for use in policy refinement and specification learning.

6. Theoretical Results

Theorem 1 (Finite-Sample Guarantees for STL-CIRL) (Joint convergence of causal formula refinement and policy learning)

Let $\mathcal{M} = (S, A, P, R, \gamma)$ be an MDP, where S is the state space with cardinality $|S|$, A is the action space with cardinality $|A|$, P is the transition kernel, R is the reward function bounded by $r_{max} = \max\{e^{\beta\rho_{max}}, -e^{-\beta\rho_{min}}\}$, and $\gamma \in [0, 1)$ is the discount factor. Then, for any $\delta \in (0, 1)$ and number of episodes K, with probability at least $1-\delta$, the following guarantees hold simultaneously (The term $1 - \delta$ represents the confidence level of the probabilistic guarantee. In probabilistic analysis, δ is a small positive number that indicates the probability of failure or the event not occurring. Therefore, $1 - \delta$ is the probability that the event will occur, which is the confidence level.): 1. Q-Learning Convergence: The learned Q-function converges to the optimal Q-function

Algorithm 3 Counterexample Generation

Require: Current state s_t, action a_t, formulas ϕ_c, ϕ_e, perturbation range ϵ
Ensure: Set of counterexamples $\mathcal{CE}$

```
Initialize CE ← ∅
τ_base ← GetCurrentTrajectory()
if ρ(φ_e, τ_base, t) ≤ 0 then
    for all state variable v_i in s_t do
        for all δ ∈ {−ε, ε} do
            s'_t ← PerturbState(s_t, v_i, δ)
            τ' ← SimulateTrajectory(s'_t, a_t)
            if IsValidCounterexample(τ', φ_c, φ_e) {Checks if τ' satisfies φ_c but violates φ_e} then
                CE ← CE ∪ {τ'}
            end if
        end for
    end for
end if
return CE
```

for the current cause formula with error bounded by $\|Q_K - Q^*(\phi_c^K)\|_\infty \leq \frac{r_{max}}{(1-\gamma)^2}\sqrt{\frac{2\log(2/\delta)}{K}}$. *This bound quantifies how quickly the agent learns the optimal policy given the current causal understanding.*

2. Formula Optimization: The objective function value of the learned cause formula approaches that of the optimal formula: $J(\phi_c^K) \geq J(\phi_c^*) - O\left(\sqrt{\frac{\log(K)}{K}}\right)$. *This guarantee ensures that our causal formula refinement process converges to an optimal explanation of the environment's dynamics.*

3. Policy Performance: The probability of satisfying the effect formula increases with training: $\mathbb{P}(\rho(\phi_e, \tau_K) > 0) \geq p^* - O(1/\sqrt{K})$, *where* p^* *is the maximum achievable satisfaction probability under any policy. This bound is derived from the concentration inequality* $|\hat{p}_K - p^*| \leq \sqrt{\frac{\log(3/\delta)}{2K}}$, *which holds with probability* $1 - \delta/3$. *Here,* $\hat{p}_K$ *represents the empirical satisfaction probability, and the concentration around* p^* *ensures that our learned policy approaches optimal performance as* K *increases. The rate of convergence is governed by both the number of episodes* K *and our confidence parameter* δ, *while being supported by our bounded robustness assumption (A4) and sufficient exploration guarantee (A1).*

Assumptions: *The theorem assumes (A1) Sufficient Exploration: Each state-action pair is visited* $\Omega(\log(K)/\epsilon^2)$ *times during training; (A2) Bounded Rewards: All rewards are bounded by* $[-r_{max}, r_{max}]$*; (A3) Kernel Regularity: The GP kernel is Lipschitz continuous with constant* L*; (A4) Bounded Robustness: The robustness values* $\rho(\phi, \tau, t)$ *are bounded for all formulas* ϕ *and trajectories* τ *(see Appendix 9.2).*

Proof We prove each claim through careful analysis of the learning dynamics:

1. **Q-learning Convergence:** For finite episodes K and bounded rewards $|r_t| \leq r_{\max}$, we apply the standard Q-learning analysis with bounded rewards. The key insight is that our exponential reward transformation maintains boundedness while emphasizing the importance of satisfying temporal logic constraints. The error bound:

$$\|Q_K - Q^*(\phi_c^K)\|_\infty \leq \frac{r_{\max}}{(1-\gamma)^2}\sqrt{\frac{2\log(2/\delta)}{K}} \tag{10}$$

follows from the Hoeffding inequality applied to the Q-learning updates, where the $(1-\gamma)^2$ term accounts for reward propagation through time and $r_{\max}$ captures the scale of our transformed rewards (see Appendix 9.4).

2. **Formula Optimization:** The Gaussian Process optimization of causal formulas achieves the following regret bound:

$$J(\phi_c^K) \geq J(\phi_c^*) - O\left(\sqrt{\frac{\beta_K \gamma_K}{K}}\right) \tag{11}$$

with probability $1 - \delta/3$. Here, $\beta_K = O(\log K)$ is the exploration bonus and γ_K is the maximum information gain of the GP model. This bound leverages the smoothness of our objective function induced by the Lipschitz kernel (A3). The probability term arises from applying the union bound across the three events (Q-learning convergence, formula optimization, and effect satisfaction), ensuring all bounds hold simultaneously with probability at least $1 - \delta$ (See Appendix 9.4).

3. **Policy Performance Bound:** The probability bound for policy performance concentrates around its true value according to Hoeffding's inequality:

$$|\hat{p}_K - p^*| \leq \sqrt{\frac{\log(3/\delta)}{2K}} \tag{12}$$

with probability $1 - \delta/3$. This bound quantifies how quickly the learned policy approaches optimal performance in terms of satisfying the effect formula, supported by our bounded robustness assumption (A4) and sufficient exploration guarantee (A1) (See Appendix 9.4). The proof demonstrates joint convergence of Q-learning and causal discovery through GP optimization, while maintaining probabilistic guarantees on task completion. ■

6.1. Existence Robustness as a Lower Bound for Formula Refinement

Theorem 2 (Existence Robustness Bound) *Let ϕ_c be a candidate cause formula, E be the existence measure, and $\rho(\tau, \phi_c, t)$ denote the robustness of ϕ_c on a trajectory τ at time t. For a set of counterexamples $\mathcal{CE}$, the existence measure E satisfies:*

$$E \leq e^{-\min_{\tau \in \mathcal{CE}} \rho(\tau, \phi_c, t)}, \tag{13}$$

where $\mathcal{CE}$ is the set of counterexamples identified during reinforcement learning exploration.

Proof The existence measure E is defined as:

$$E = e^{-\text{Mean}(\rho(\tau, \phi_c, t))}, \quad \forall \tau \in \mathcal{CE}, \tag{14}$$

where $\text{Mean}(\rho(\tau, \phi_c, t))$ denotes the average robustness of ϕ_c over all counterexamples $\tau \in \mathcal{CE}$.

By definition of the mean, it holds that:

$$\text{Mean}(\rho(\tau, \phi_c, t)) \geq \min_{\tau \in \mathcal{CE}} \rho(\tau, \phi_c, t). \tag{15}$$

Since the exponential function e^{-x} is monotonically decreasing with respect to x, we have:

$$e^{-\text{Mean}(\rho(\tau, \phi_c, t))} \leq e^{-\min_{\tau \in \mathcal{CE}} \rho(\tau, \phi_c, t)}. \tag{16}$$

Substituting the definition of E into this inequality, it follows that:

$$E \leq e^{-\min_{\tau \in \mathcal{CE}} \rho(\tau, \phi_c, t)}. \tag{17}$$

This completes the proof of the bound. ■

7. Implementation and Experiments

7.1. Case Study: Gene Regulation Environment

To evaluate our approach, we implemented a gene regulation environment where an RL agent discovers causal relationships between gene mutations and disease progression. The environment consists of a 5×5 grid where the agent interacts with four genes: G_1, G_2, G_3, and G_4. The state space is $S = \{(x, y), G_1, G_2, G_3, G_4, D\}$, where (x, y) is the agent's position, $G_i \in \{0, 1\}$ is gene i's mutation status, and $D \in [0, 100]$ is the disease progression level. In this context, disease progression occurs when genes G_1, G_2, and G_4 are in their mutated state (value of 1), establishing the underlying causal mechanism that the agent must discover to effectively manage the disease. The action space $\mathcal{A}$ includes movement actions (UP, DOWN, LEFT, RIGHT) and gene modification actions for each gene. The environment's underlying causal structure is represented by the following Causal STL formula:

$$\begin{aligned}\Phi := \text{do}\Big(&\Box_{[0,10]}(G_1 = 1 \wedge G_2 = 1 \wedge G_4 = 1 \wedge G_3 = 0) \wedge \\ &\Diamond_{[0,3]}(\text{ModifyG1} = 0) \wedge \Diamond_{[3,6]}(\text{ModifyG2} = 0) \wedge \Diamond_{[6,9]}(\text{ModifyG4} = 0)\Big) \\ &\rightsquigarrow \Diamond_{[12,15]}(\text{DiseaseProgression} = 0)\end{aligned} \tag{18}$$

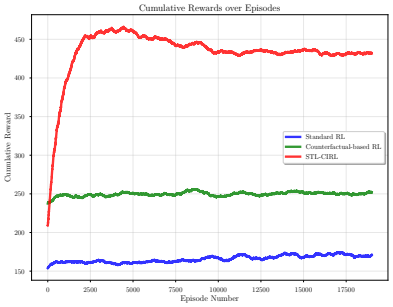

Figure 1: Performance comparison between Standard RL, Counterfactual-based RL, and STL-CIRL approaches in the gene regulation environment.

where t_1, t_2, and t_3 represent temporal bounds for modification steps, and δ represents the time window for disease progression to reach zero. Results show STL-CIRL achieved faster learning and superior performance compared to baseline and Counterfactual-based RL approaches.

7.2. Case Study 2: Traffic Signal Control

To evaluate our approach in another domain, we implemented a traffic control environment with three intersections in a row, each with one perpendicular crossing street. Vehicles need to travel from point A to point B along the main horizontal road, with the agent controlling traffic signals to minimize travel time while managing cross-traffic. The state space S is defined as $S = \{(QH_i, QV_i, T) \mid i \in \{1, 2, 3\}\}$ where $QH_i \in \{0, 1, 2\}$ represents the horizontal queue length, $QV_i \in \{0, 1, 2\}$ is the vertical queue length, and T is the travel time from point A to point B. These queue values correspond to low (0), medium (1), and high (2) traffic congestion levels at each intersection. The action space $\mathcal{A}$ includes traffic signal phase changes that control vehicle flow at each

intersection. The environment's causal structure is formalized through the following Causal STL formula, where $i \in \{1, 2, 3\}$ represents the three intersections:

$$\begin{aligned}\Phi := \text{do}\Big(&\Box_{[0,3]}(QH_i \geq 2 \wedge QV_i < 1) \wedge \\ &\Diamond_{[0,1]}(\text{SignalChange}_1 = 1) \wedge \Diamond_{[1,2]}(\text{SignalChange}_2 = 1) \wedge \Diamond_{[2,3]}(\text{SignalChange}_3 = 1)\Big) \\ &\rightsquigarrow \Diamond_{[4,5]}(T < 30)\end{aligned} \tag{19}$$

This formula expresses that when the horizontal queue length exceeds a threshold value while the vertical queue remains below another threshold (indicating a need to prioritize the main road), changing the signal sequentially at each intersection should eventually reduce the travel time from point A to B below a target threshold.

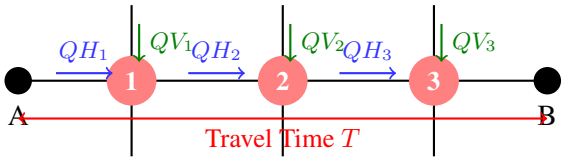

Figure 2: Traffic control environment with three intersections. The agent manages horizontal (QH_i) and vertical (QV_i) queue lengths to minimize travel time from A to B.

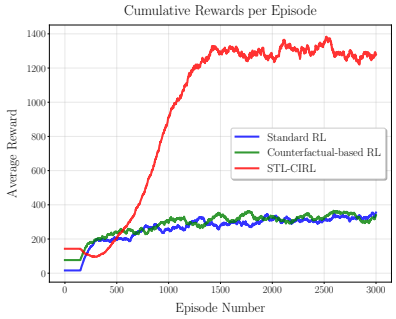

Figure 3: Performance comparison between Standard RL, Counterfactual-based RL, and STL-CIRL approaches in the traffic control environment.

8. Conclusion

We introduced STL-CIRL, a framework combining Causal Signal Temporal Logic with reinforcement learning. Our contributions include a method for extracting causal temporal logic formulas from RL data, theoretical convergence guarantees with sample complexity bounds, and dynamic counterfactual traces. Experiments in gene regulation and traffic control show our approach consistently outperforms conventional RL methods. These findings highlight the benefits of causal reasoning with temporal logic in RL, pointing to future work in stochastic and multi-agent settings.

Acknowledgments

This work is partially supported by NSF CNS 2304863, CNS 2339774, IIS 2332476, and ONR N00014-23-1-2505.

References

Derya Aksaray, Austin Jones, Zhaodan Kong, Mac Schwager, and Calin Belta. Q-learning for robust satisfaction of signal temporal logic specifications. *arXiv preprint arXiv:1609.07409*, 2016.

Shayan Meshkat Alsadat, Nasim Baharisangari, Yash Paliwal, and Zhe Xu. Distributed reinforcement learning for swarm systems with reward machines. In *2024 American Control Conference (ACC)*, pages 33–38, 2024. doi: 10.23919/ACC60939.2024.10644549.

Elias Bareinboim. Towards causal reinforcement learning. *Proceedings of the 37th International Conference on Machine Learning*, 2020. URL https://crl.causalai.net/crl-icml20.pdf.

D. Bertsekas and J.N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996. ISBN 9781886529106. URL https://books.google.com/books?id=txw6EAAAQBAJ.

Paul Cabilio. Sequential estimation in bernoulli trials. *The Annals of Statistics*, 5(2), March 1977. ISSN 0090-5364. doi: 10.1214/aos/1176343799. URL http://dx.doi.org/10.1214/aos/1176343799.

Emile Contal, Vianney Perchet, and Nicolas Vayatis. Gaussian process optimization with mutual information. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 253–261, Bejing, China, 22–24 Jun 2014. PMLR. URL https://proceedings.mlr.press/v32/contal14.html.

Jan Corazza, Hadi Partovi Aria, Daniel Neider, and Zhe Xu. Expediting reinforcement learning by incorporating knowledge about temporal causality in the environment. In Francesco Locatello and Vanessa Didelez, editors, *Proceedings of the Third Conference on Causal Learning and Reasoning*, volume 236 of *Proceedings of Machine Learning Research*, pages 643–664. PMLR, 01–03 Apr 2024. URL https://proceedings.mlr.press/v236/corazza24a.html.

Ishita Dasgupta, Jane Wang, Silvia Chiappa, Jovana Mitrovic, Pedro Ortega, David Raposo, Edward Hughes, Peter Battaglia, Matthew Botvinick, and Zeb Kurth-Nelson. Causal reasoning from meta-reinforcement learning, 2019. URL https://arxiv.org/abs/1901.08162.

Ziquan Deng, Samuel P. Eshima, James Nabity, and Zhaodan Kong. Causal signal temporal logic for the environmental control and life support system's fault analysis and explanation. *IEEE Access*, 11:26471–26482, 2023. doi: 10.1109/ACCESS.2023.3246512.

Wenhao Ding, Haohong Lin, Bo Li, and Ding Zhao. Generalizing goal-conditioned reinforcement learning with variational causal reasoning, 2023. URL https://arxiv.org/abs/2207.09081.

Jianqing Fan, Bai Jiang, and Qiang Sun. Hoeffding's inequality for general markov chains and its applications to statistical learning. *Journal of Machine Learning Research*, 22(139):1–35, 2021. URL http://jmlr.org/papers/v22/19-479.html.

Christian Fiedler. Lipschitz and hölder continuity in reproducing kernel hilbert spaces, 2023. URL https://arxiv.org/abs/2310.18078.

Susmit Jha, Ashish Tiwari, Sanjit A. Seshia, Tuhin Sahai, and Natarajan Shankar. Telex: learning signal temporal logic from positive examples using tightness metric. *Formal Methods in System Design*, 54(3):364–387, January 2019. ISSN 1572-8102. doi: 10.1007/s10703-019-00332-1. URL http://dx.doi.org/10.1007/s10703-019-00332-1.

Ming Jin and Javad Lavaei. Stability-certified reinforcement learning: A control-theoretic perspective, 2018. URL https://arxiv.org/abs/1810.11505.

Xiao Li, Cristian-Ioan Vasile, and Calin Belta. Reinforcement learning with temporal logic rewards, 2017. URL https://arxiv.org/abs/1612.03471.

Matthias Seeger. Gaussian processes for machine learning. *International Journal of Neural Systems*, 14(02):69–106, April 2004. ISSN 1793-6462. doi: 10.1142/s0129065704001899. URL http://dx.doi.org/10.1142/s0129065704001899.

Niranjan Srinivas, Andreas Krause, Sham M. Kakade, and Matthias W. Seeger. Information-theoretic regret bounds for gaussian process optimization in the bandit setting. *IEEE Transactions on Information Theory*, 58(5):3250–3265, 2012. doi: 10.1109/TIT.2011.2182033.

Sattar Vakili, Nacime Bouziani, Sepehr Jalali, Alberto Bernacchia, and Da shan Shiu. Optimal order simple regret for gaussian process bandits, 2021. URL https://arxiv.org/abs/2108.09262.

Harish Venkataraman, Derya Aksaray, and Peter Seiler. Tractable reinforcement learning of signal temporal logic objectives. In Alexandre M. Bayen, Ali Jadbabaie, George Pappas, Pablo A. Parrilo, Benjamin Recht, Claire Tomlin, and Melanie Zeilinger, editors, *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, volume 120 of *Proceedings of Machine Learning Research*, pages 308–317. PMLR, 10–11 Jun 2020. URL https://proceedings.mlr.press/v120/venkataraman20a.html.

Justin Whitehouse, Aaditya Ramdas, and Steven Z. Wu. On the sublinear regret of gp-ucb. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 35266–35276. Curran Associates, Inc., 2023.

Yunfeng Zhang, Jaehyon Paik, and Peter Pirolli. Reinforcement learning and counterfactual reasoning explain adaptive behavior in a changing environment. *Topics in Cognitive Science*, 7(2):368–381, April 2015. ISSN 1756-8765. doi: 10.1111/tops.12143. URL http://dx.doi.org/10.1111/tops.12143.

9. Appendix

9.1. Counterfactual-based Reinforcement Learning

For comparison purposes, we implement a Counterfactual Reinforcement Learning (CF-RL) agent that utilizes counterfactual reasoning without structured causal knowledge, building on work by (Zhang et al. (2015)). This approach creates counterfactual states through a parametric transformation $s'_t = f(s_t, a_t, \theta)$, where θ denotes environmental parameters. The agent optimizes a composite

reward function that combines observed and counterfactual outcomes:

$$R_{\text{total}} = (1-\lambda)R_{\text{actual}} + \lambda R_{\text{cf}} \tag{20}$$

where $\lambda \in [0,1]$ weights the counterfactual influence. Counterfactual rewards incorporate a similarity-weighted function:

$$R_{\text{cf}}(s'_t, a_t) = R_{\text{actual}}(s'_t, a_t) \cdot \exp\left(-\frac{\|s_t - s'_t\|^2}{2\sigma^2}\right) \tag{21}$$

where σ controls the influence radius of counterfactual states. The Q-values update incorporates both actual and counterfactual experiences:

$$Q(s_t, a_t) \leftarrow (1-\alpha)Q(s_t, a_t) + \alpha\left[r_t + \gamma \max_{a'} Q(s_{t+1}, a') + \beta R_{\text{cf}}\right] \tag{22}$$

with learning rate α, discount factor γ, and counterfactual weight β.

9.2. Robustness Calculation and Bounds

The robustness degree for STL formulas, for any Signal s, is calculated recursively according to the following rules (Aksaray et al. (2016)):

$$\begin{aligned}
\rho(s, \neg(f(s) < d), t) &= -\rho(s, (f(s) < d), t) \\
\rho(s, (f(s) < d), t) &= d - f(s_t) \\
\rho(s, \phi_1 \wedge \phi_2, t) &= \min(\rho(s, \phi_1, t), \rho(s, \phi_2, t)) \\
\rho(s, \phi_1 \vee \phi_2, t) &= \max(\rho(s, \phi_1, t), \rho(s, \phi_2, t)) \\
\rho(s, \Box_{[a,b]}\phi, t) &= \min_{t' \in [t+a, t+b]} \rho(s, \phi, t') \\
\rho(s, \Diamond_{[a,b]}\phi, t) &= \max_{t' \in [t+a, t+b]} \rho(s, \phi, t')
\end{aligned} \tag{23}$$

From these calculations, we can derive the following bounds:

Theorem 3 (Robustness Bounds) *For an STL formula ϕ and signal s, the robustness degree is bounded as follows:*

1. For atomic predicates:

$$-M \leq \rho(s, (f(s) < d), t) \leq d \tag{24}$$

where $M = \sup_t |f(s_t)|$ is the supremum of the signal values.

2. For Boolean combinations:

$$\begin{aligned}
\min(\rho_{min}(\phi_1), \rho_{min}(\phi_2)) &\leq \rho(s, \phi_1 \wedge \phi_2, t) \\
&\leq \min(\rho_{max}(\phi_1), \rho_{max}(\phi_2))
\end{aligned} \tag{25}$$

$$\begin{aligned}
\max(\rho_{min}(\phi_1), \rho_{min}(\phi_2)) &\leq \rho(s, \phi_1 \vee \phi_2, t) \\
&\leq \max(\rho_{max}(\phi_1), \rho_{max}(\phi_2))
\end{aligned} \tag{26}$$

3. For temporal operators:

$$\rho_{min}(\phi) \leq \rho(s, \Box_{[a,b]}\phi, t) \leq \rho_{max}(\phi) \tag{27}$$

$$\rho_{min}(\phi) \leq \rho(s, \Diamond_{[a,b]}\phi, t) \leq \rho_{max}(\phi) \tag{28}$$

9.3. Foundation Theorems

The bounds in our main theorem build upon several fundamental results from reinforcement learning and optimization theory:

Theorem 4 (STL Robustness Properties) *For an STL formula ϕ and trajectories τ_1, τ_2, the robustness degree $\rho(\tau, \phi, t)$ satisfies (Jha et al. (2019); Aksaray et al. (2016)):*

1. ***Soundness:*** $\rho(\tau, \phi, t) > 0 \implies \tau \models \phi$ *at time* t
2. ***Completeness:*** $\tau \models \phi$ *at time* $t \implies \rho(\tau, \phi, t) \geq 0$
3. ***Lipschitz Continuity:*** *For any two trajectories τ_1, τ_2:*

$$|\rho(\tau_1, \phi, t) - \rho(\tau_2, \phi, t)| \leq L_\phi \|\tau_1 - \tau_2\|_\infty \tag{29}$$

   where L_ϕ is the Lipschitz constant of ϕ.

4. ***Compositional Bounds:*** *For temporal operators:*

$$\begin{aligned} \rho(\tau, \Diamond_{[a,b]}\phi, t) &\leq \max_{t' \in [t+a, t+b]} \rho(\tau, \phi, t') \\ \rho(\tau, \Box_{[a,b]}\phi, t) &\geq \min_{t' \in [t+a, t+b]} \rho(\tau, \phi, t') \end{aligned} \tag{30}$$

These properties ensure that robustness degrees provide meaningful quantitative measures of satisfaction and enable stable learning dynamics.

Proof 1. **Soundness:** By construction of the robustness degree, $\rho(\tau, \phi, t) > 0$ implies that τ satisfies ϕ with a positive margin, ensuring satisfaction.

2. **Completeness:** If $\tau \models \phi$, the satisfaction must occur with some non-negative margin, thus $\rho(\tau, \phi, t) \geq 0$.

3. **Lipschitz Continuity:** For atomic predicates μ, the result follows from the Lipschitz continuity of the predicates themselves. For temporal operators, we apply the triangle inequality and use induction on the formula structure.

4. **Compositional Bounds:** These follow directly from the semantics of eventually ($\Diamond$) and always ($\Box$) operators. Eventually takes the maximum robustness over the interval, while always takes the minimum robustness over the interval. The result follows from the definition of temporal operators and the monotonicity of min/max operations.

■

Theorem 5 (Hoeffding's Inequality) *Let $X_1, \ldots, X_n$ be independent random variables with $a_i \leq X_i \leq b_i$ for each i. Let $\bar{X} = \frac{1}{n}\sum_{i=1}^n X_i$. Then for any $t > 0$:*

$$\mathbb{P}(|\bar{X} - \mathbb{E}[\bar{X}]| \geq t) \leq 2\exp\left(-\frac{2n^2t^2}{\sum_{i=1}^n (b_i - a_i)^2}\right) \tag{31}$$

This inequality is crucial for establishing our Q-learning convergence bounds (Fan et al. (2021)).

Theorem 6 (GP-UCB Regret Bound) *For a GP-UCB algorithm with kernel k and noise variance σ^2, after T rounds, with probability at least $1-\delta$, the cumulative regret R_T is bounded by (Srinivas et al. (2012)):*

$$R_T \leq \sqrt{C_1 T \beta_T \gamma_T} \tag{32}$$

where $\beta_T = 2\log(|\mathcal{A}|T^2\pi^2/(6\delta))$, γ_T is the maximum information gain, and C_1 is a constant depending on the kernel.

Theorem 7 (Bellman Contraction) *For any two Q-functions Q_1 and Q_2, the Bellman operator $\mathcal{T}$ is a contraction in the sup-norm (Bertsekas and Tsitsiklis (1996)):*

$$\|\mathcal{T}Q_1 - \mathcal{T}Q_2\|_\infty \leq \gamma \|Q_1 - Q_2\|_\infty \tag{33}$$

where $\gamma \in [0,1)$ is the discount factor. This guarantees the convergence of Q-learning.

9.4. Mathematical Foundations and Derivations

The mathematical equations and inequalities in our theoretical results arise from several key principles:

1. Q-Learning Error Bound: The Q-learning error bound

$$\|Q_K - Q^*(\phi_c^K)\|_\infty \leq \frac{r_{\max}}{(1-\gamma)^2}\sqrt{\frac{2\log(2/\delta)}{K}} \tag{34}$$

emerges from a rigorous analysis of the learning dynamics. The derivation proceeds as follows:

a) Value Propagation Analysis: The geometric series of discounted rewards yields:

$$\sum_{t=0}^{\infty} \gamma^t r_{\max} = \frac{r_{\max}}{1-\gamma} \tag{35}$$

This sum represents the maximum possible cumulative reward, where $r_{\max}$ bounds individual rewards. The factor $\frac{1}{1-\gamma}$ accounts for infinite-horizon discounting.

b) Concentration Inequality: By Hoeffding's inequality, for any $\epsilon > 0$:

$$\mathbb{P}(|\hat{Q}_K(s,a) - Q^*(s,a)| \geq \epsilon) \leq 2\exp\left(-\frac{2K\epsilon^2}{r_{\max}^2}\right) \tag{36}$$

where $\hat{Q}_K$ is the empirical Q-function after K episodes.

c) Error Propagation: The combined error analysis yields:

$$\|Q_{k+1} - Q^*\|_\infty \leq \gamma\|Q_k - Q^*\|_\infty + \frac{r_{\max}}{1-\gamma}\sqrt{\frac{2\log(2/\delta)}{k}} \tag{37}$$

Through telescoping and taking the limit as $k \to K$, we obtain our final bound. The result captures the effect of finite sampling through $O(1/\sqrt{K})$ convergence, accounts for reward propagation via the $(1-\gamma)^2$ term, provides high-probability guarantees through $\log(2/\delta)$, and maintains tight dependence on the reward scale $r_{\max}$.

2. Formula Optimization Bound: The bound $J(\phi_c^K) \geq J(\phi_c^*) - O\left(\sqrt{\frac{\beta_K \gamma_K}{K}}\right)$ provides a performance guarantee for the optimization process, showing how close the algorithm gets to the optimal objective value $J(\phi_c^*)$ after K iterations. This bound arises from the interaction of several foundational principles in Gaussian Process (GP) optimization, detailed as follows:

a) GP-UCB Regret Analysis: The cumulative regret R_T captures the total performance gap between the optimal choice and the selected points over T iterations (Whitehouse et al. (2023)):

$$R_T = \sum_{t=1}^{T} \big(f(x^*) - f(x_t)\big), \tag{38}$$

where $f(x^*)$ is the value at the optimal point, and $f(x_t)$ is the value at the selected point at time t. This represents the "cost of learning" due to exploration. The regret accumulates as the algorithm balances exploration (gathering information about f) and exploitation (selecting high-performing points).

b) Information Gain Analysis: The maximum information gain γ_T quantifies the reduction in uncertainty about the function f over time (Vakili et al. (2021)):

$$\gamma_T = \frac{1}{2} \log |I + \sigma^{-2} K_T|, \tag{39}$$

where I is the identity matrix, σ^2 is the noise variance, K_T is the kernel matrix of the GP model at time T, and $|\cdot|$ denotes the determinant. This term measures how much knowledge the algorithm has gained about the objective function through the collected observations. The kernel matrix encodes the correlations between points in the input space, allowing the GP to interpolate and reduce uncertainty.

c) Kernel Regularity Property: The Lipschitz continuity of the kernel function guarantees smooth interpolation of the GP model (Fiedler (2023)):

$$|k(x, x') - k(y, y')| \leq L\big(\|x - y\| + \|x' - y'\|\big), \tag{40}$$

where L is the Lipschitz constant. This property ensures that similar inputs produce similar outputs, that the objective function f does not change abruptly in small neighborhoods, and that predictions of the GP model are reliable around observed data points. Smoothness is critical for ensuring stable convergence and accurate predictions during optimization.

d) RBF Kernel Information Gain: For the commonly used Radial Basis Function (RBF) kernel, the maximum information gain is bounded as (Srinivas et al. (2012)):

$$\gamma_T = O((\log T)^{d+1}), \tag{41}$$

where d is the input dimension. This bound implies that information gain grows logarithmically with iterations T, ensuring efficient exploration. It scales polynomially with input dimension, making it suitable for moderately high-dimensional problems, while limiting the computational cost of updating the GP model.

Integration of Bounds for Formula Optimization: The bound combines key insights to guarantee predictable convergence to the optimal formula:

1. **Bounded Robustness Contribution:** For two parameterized STL formulas $\phi_1 = \phi(\theta_1)$ and $\phi_2 = \phi(\theta_2)$, the change in robustness for a trajectory τ at time t is bounded by:

$$|\rho(\tau, \phi(\theta_1), t) - \rho(\tau, \phi(\theta_2), t)| \leq L_\rho \|\theta_1 - \theta_2\|, \tag{42}$$

   where $\|\theta_1 - \theta_2\|$ is the distance between parameter vectors, and L_ρ is the Lipschitz constant for robustness with respect to the parameters. This property ensures that small changes in parameters lead to predictable changes in robustness, stabilizing the optimization process.

2. **Information Gain Accumulation:** The accumulated information gain reduces uncertainty over time, contributing to faster convergence (Contal et al. (2014)):

$$\gamma_K = \sum_{t=1}^{K} I(y_t; f_t | \mathcal{D}_{t-1}) = O((\log K)^{d+1}), \tag{43}$$

   where $I(y_t; f_t|\mathcal{D}_{t-1})$ is the mutual information between observations y_t and the function f_t given the past data.

3. **Posterior Variance Reduction:** The GP posterior variance decreases as more observations are made (Contal et al. (2014)):

$$\sigma_K^2(x) \leq \frac{\beta_K \gamma_K}{K}, \tag{44}$$

   where $\beta_K = O(\log K)$ is the exploration parameter. This reduction reflects increased confidence in the GP model's predictions as K grows.

4. **Combined Optimization Bound:** Together, these components yield the final bound:

$$J(\phi_c^K) \geq J(\phi_c^*) - \sqrt{\frac{2\beta_K \gamma_K}{K}}, \tag{45}$$

   This final bound emerges from the following reasoning: The cumulative regret analysis provides the basic $O(1/\sqrt{K})$ convergence rate. The information gain γ_K moderates exploration efficiency through uncertainty reduction. The exploration parameter β_K ensures sufficient exploration while maintaining exploitation. The Lipschitz continuity of the kernel guarantees smooth interpolation between observations. The square root form arises because the posterior variance $\sigma_K^2(x)$ contributes quadratically to the uncertainty. The exploration-exploitation tradeoff requires balancing immediate rewards with information gain. The cumulative regret accumulates as $\sqrt{K}$ due to the martingale property of the GP model.

3. Policy Performance Bound: The probability bound

$$\mathbb{P}(\rho(\tau_K, \phi_e, t) > 0) \geq p^* - O(1/\sqrt{K}) \tag{46}$$

follows from several key theoretical components that together establish the convergence rate of policy performance:

1. **Empirical Bernoulli Estimation:** The empirical success probability $\hat{p}_K$ is estimated as (Cabilio (1977)):

$$\hat{p}_K = \frac{1}{K}\sum_{i=1}^{K} \mathbb{1}[\rho(\tau_i, \phi_e, t) > 0] \tag{47}$$

where $\mathbb{1}[\cdot]$ is the indicator function that equals 1 if the condition is true and 0 otherwise. This estimates the fraction of trajectories that satisfy the effect formula by evaluating the ratio of successful trajectories (those with positive robustness) to the total number of trajectories K, effectively converting continuous robustness values into binary satisfaction outcomes.

2. **Concentration Analysis:** By Hoeffding's inequality for bounded random variables:

$$\mathbb{P}(|\hat{p}_K - p^*| \geq \epsilon) \leq 2\exp(-2K\epsilon^2) \tag{48}$$

This inequality bounds the probability that our empirical estimate $\hat{p}_K$ deviates from the true probability p^* by more than ϵ. Setting $\epsilon = \sqrt{\frac{\log(3/\delta)}{2K}}$ yields our desired confidence level.

3. **Bounded Robustness:** By assumption (A4), robustness values are bounded:

$$|\rho(\tau, \phi_e, t)| \leq M \text{ for some } M > 0 \tag{49}$$

This boundedness is crucial as it ensures the validity of concentration inequalities, stabilizes learning dynamics, and enables meaningful probability estimates.

Integration of Components: These elements combine to establish the policy performance bound through the following logic:

1. The empirical estimation provides a consistent estimator of satisfaction probability. 2. Hoeffding's inequality quantifies the estimation error rate as $O(1/\sqrt{K})$. 3. Bounded robustness fundamentally guarantees stable learning dynamics through several mechanisms:

a) Gradient Stability: Bounded robustness implies that the robustness measure $\rho(\phi_e, \tau, t)$ is both:

1. **Lipschitz in the parameters θ:** There exists L_ρ such that

$$|\rho(\theta + \Delta\theta) - \rho(\theta)| \leq L_\rho \, \|\Delta\theta\|. \tag{50}$$

2. **Bounded by M:** The function $\rho(\phi_e, \tau, t)$ (or its range) does not exceed M in absolute value.

From these two assumptions, it follows that the gradient of ρ w.r.t. θ is also bounded:

$$\|\nabla_\theta \rho(\tau, \phi_e, t)\| \;\leq\; L_\rho \, M. \tag{51}$$

This result prevents exploding gradients during learning by ensuring gradient updates remain within controlled limits (Jin and Lavaei (2018)).

b) Value Function Convergence: Since $\rho(\phi_e, \tau, t)$ is bounded by M, any Q-function update tied to ρ changes by at most γM at each iteration, where γ is the discount factor:

$$|Q_{t+1}(s, a) - Q_t(s, a)| \;\leq\; \gamma \, M. \tag{52}$$

This cap on the change in $Q_t(s, a)$ values enforces stable value iteration, preventing excessive swings from one iteration to the next.

c) Policy Update Stability: Since the robustness $\rho(\phi_e, \tau, t)$ is bounded by M, this translates into a limit on how much the associated Q-values (or value function) can change. Consequently, the induced policy changes are also constrained. A key observation is that a policy π is a probability distribution over actions, and when one distribution π_{t+1} shifts probability mass from an action a to another action b, the ℓ_∞ difference $\|\pi_{t+1}(\cdot) - \pi_t(\cdot)\|_\infty$ can, in the worst case, incur a factor of 2 (moving probability mass from 0 to 1 for one action and from 1 to 0 for another). Combining this with the bounded change in Q-values from iteration to iteration yields:

$$\|\pi_{t+1} - \pi_t\|_\infty \;\leq\; \frac{2M}{1-\gamma} \tag{53}$$

which ensures that the policy does not shift too abruptly. This factor of 2 accounts for the possibility of moving all probability mass from one action to another, and the term $(1-\gamma)$ in the denominator reflects the discounting in reinforcement learning, leading to stable and gradual learning progress.

The final bound emerges from:

$$\begin{aligned} \mathbb{P}(\rho(\tau_K, \phi_e, t) > 0) &= p^* + (\hat{p}_K - p^*) \\ &\geq p^* - |\hat{p}_K - p^*| \\ &\geq p^* - O(1/\sqrt{K}) \end{aligned} \tag{54}$$

This shows that the probability of satisfying the effect formula approaches the optimal probability p^* at a rate of $O(1/\sqrt{K})$, which is optimal for statistical estimation without additional smoothness assumptions.