Reinforcement Learning with Temporal-Logic-Based Causal Diagrams

Yash Paliwal[⋆1], Rajarshi Roy[*2], Jean-Raphaël Gaglione[*3], Nasim Baharisangari[1], Daniel Neider[4,5], Xiaoming Duan[6], Ufuk Topcu[3], and Zhe Xu[1]

[1] Arizona State University, Arizona, USA
[2] Max Planck Institute for Software Systems, Kaiserslautern, Germany
[3] University of Texas at Austin, Texas, USA
[4] TU Dortmund University, Dortmund, Germany
[5] Center for Trustworthy Data Science and Security, University Alliance Ruhr, Germany
[6] Department of Automation, Shanghai Jiao Tong University

Abstract. We study a class of reinforcement learning (RL) tasks where the objective of the agent is to accomplish temporally extended goals. In this setting, a common approach is to represent the tasks as deterministic finite automata (DFA) and integrate them into the state-space for RL algorithms. However, while these machines model the reward function, they often overlook the causal knowledge about the environment. To address this limitation, we propose the Temporal-Logic-based Causal Diagram (TL-CD) in RL, which captures the temporal causal relationships between different properties of the environment. We exploit the TL-CD to devise an RL algorithm in which an agent requires significantly less exploration of the environment. To this end, based on a TL-CD and a task DFA, we identify configurations where the agent can determine the expected rewards early during an exploration. Through a series of case studies, we demonstrate the benefits of using TL-CDs, particularly the faster convergence of the algorithm to an optimal policy due to reduced exploration of the environment.

Keywords: Reinforcement Learning · Causal Inference · Neuro-Symbolic AI.

1 Introduction

In many reinforcement learning (RL) tasks, the objective of the agent is to accomplish temporally extended goals that require multiple actions to achieve. One common approach to modeling these goals is to use finite state machines. However, these machines only model the reward function and do not take into account the causal knowledge of the underlying environment, which can limit the effectiveness of the RL algorithms [2, 3, 6, 17, 20, 25–28].

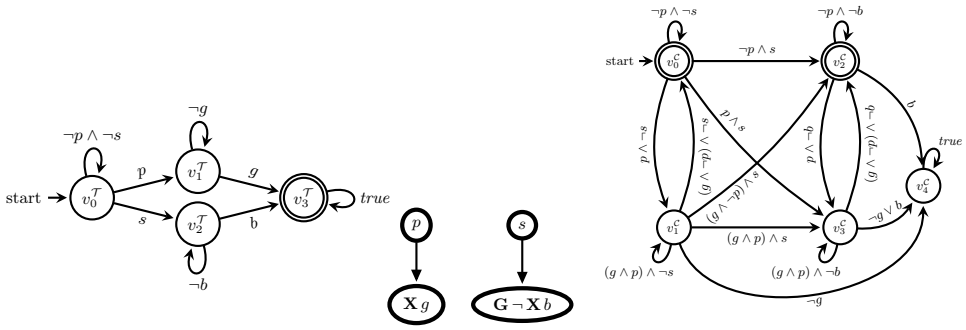
⋆ The first three authors contributed equally.

Moreover, online RL, including in the non-Markovian setting, often requires extensive interactions with the environment. This impedes the adoption of RL algorithms in real-world applications due to the impracticality of expensive and/or unsafe data collection during the exploration phase.

To address these limitations, in this paper we propose Temporal-Logic-based Causal Diagrams (TL-CDs) which can capture the temporal causal relationships between different properties of the environment, allowing the agent to make more informed decisions and require less exploration of the environment. TL-CDs combine temporal logic, which allows for reasoning about events over time, with causal diagrams, which represent the causal relationships between variables. By using TL-CDs, the RL algorithm can exploit the causal knowledge of the environment to identify configurations where the agent can determine the expected rewards early during an exploration, leading to faster convergence to an optimal policy.

We introduce an RL algorithm that leverages TL-CDs to achieve temporally extended goals. We show that our algorithm requires significantly less exploration of the environment than traditional RL algorithms that use finite state machines to model goals. By using TL-CDs, our algorithm identifies configurations where the agent can determine the expected rewards early during exploration, reducing the number of steps required to achieve the goal.

2 Motivating Example



(a) Task DFA $\mathcal{T}$ (b) TL-CD $\mathfrak{C}$ (c) DFA $\mathcal{C}$ derived from $\mathfrak{C}$

Fig. 1: The seed environment. The four propositions are p (the agent plants the seed), g (a tree grows), s (the agent sells the seed) and b (the agent buys a tree).

Let us take a running example to illustrate the concept. There is a farmer who possesses a unique seed and his objective is to obtain a tree. There are two potential ways to achieve this goal. First, the farmer can plant the seed (p) and wait for the tree to grow (g). Alternatively, the farmer can sell the seed (s) and

use the money to purchase a tree (b). The set of four propositions can thus be represented as $\mathcal{P} = \{p, g, s, b\}$. Figure 1a illustrates the corresponding task DFA $\mathcal{T}$. Additional causal information is provided with the TL-CD $\mathfrak{C}$ (Figure 1b), interpreted as follows: $p \dashrightarrow \mathbf{X}\, g$ expresses that planting a tree will result in a tree growing in the next time-step (e.g., year), and $s \dashrightarrow \mathbf{G} \neg \mathbf{X}\, b$ expresses that selling the seed leads to never being able to buy a tree (as the farmer will never find an offer for a tree that is cheaper than a seed). This TL-CD is equivalent to the *causal* DFA $\mathcal{C}$ illustrated in Figure 1c (details are provided later).

3 Related work

Causal inference answers questions about the mechanism by which manipulating one or a set of variables affects another variable or a set of variables [22]. In other words, through causal inference, we infer the cause and effect relationships among the variables from observational data, experimental data, or a combination of both [16].

Recently, the inherent capabilities of reinforcement learning (RL) and causal inference (CI) have simultaneously been used for better decision-making including both interventional reasoning [13, 14, 24, 30] and counterfactual reasoning [4,5,9,12] in different settings [15]. In other words, in an RL setting, harnessing casual knowledge including causal relationships between the actions, rewards, and intrinsic properties of the domain where the agent is deployed can improve the decision-making abilities of the agent [15].

Usually, incorporating CI in an RL setting can be done using three types of data including observational data, experimental data, and counterfactual data accompanied by the causal diagram of the RL setting, if available. An agent can have access to observational data by observing another agent, observing the environment, offline learning, acquiring prior knowledge about the underlying setting, etc. Experimental data can be acquired by actively interacting (intervening) with the environment. Counterfactual data can be generated using a specified model, estimated through active learning empirically [5, 18, 21].

In connecting CI and RL, the mentioned data types have been used by themselves or in different combinations. For example, in [10], through sampling observational data in new environments, an agent can make minimal necessary adaptions to optimize the policy given diagrams of structural relationship among the variables of the RL setting. In [29], both observational and experimental data (empirical data) are used to learn *causal states* which are the coarsest partition of the joint history of actions and observations that are maximally predictive of the future in partially observable Markov decision processes (POMDP). In [5], a combination of all data types has been used in a Multi-Armed Bandit problem in order to improve the personalized decision-making of the agent, where the effect of unmeasured variables (unobserved confounders) has been taken into consideration.

Our research is closely linked to the utilization of formal methods in reinforcement learning (RL), such as RL for reward machines [11] and RL with

temporal logic specifications [2, 3, 6, 17, 20, 25–28]. For instance, [11] proposed a technique known as Q-learning for reward machines (QRM) and demonstrated that QRM can almost certainly converge to an optimal policy in the tabular case. Additionally, QRM outperforms both Q-learning and hierarchical RL for tasks where the reward functions can be encoded by reward machines. However, none of these works have incorporated the causal knowledge in expediting the RL process.

4 Preliminaries

As typically done in RL problems, we rely on Markov Decision Processes (MDP) [23] to model the effects of sequential decisions of an RL agent. We, however, deviate slightly from the standard definition of MDPs. This is to be able to capture temporally extended goals for the agent and thus, want the reward to be non-Markovian. To capture non-Markovian rewards, we rely on simple finite state machines—deterministic finite automaton (DFA). Further, to express causal relationships in the environment, we rely on the de facto temporal logic, Linear Temporal Logic (LTL). We introduce all the necessary concepts formally in this section.

Labeled Markov Decision Process. A *labeled Markov decision process* [11] is a tuple $\mathcal{M} = \langle S, s_I, A, p, r, \mathcal{P}, L\rangle$ consisting of a finite set of states S, an agent's initial state $s_I \in S$, a finite set of actions A, a transition probability function $p : S \times A \mapsto \Delta(S)$, a non-Markovian reward function $r : (S \times A)^* \times S \mapsto \mathbb{R}$, a set of relevant propositions $\mathcal{P}$, and a labeling function $L : S \times A \times S \mapsto 2^{\mathcal{P}}$. Here $\Delta(S)$ denotes the set of all probability distributions over S. We denote by $p(s'|s, a)$ the probability of transitioning to state s' from state s under action a. Additionally, we include a set of propositions $\mathcal{P}$ that track the relevant information that the agent senses in the environment. We integrate the propositions in the labeled MDP using the labeling function L.

We define a *trajectory* to be the realization of the stochastic process defined by a labeled MDP. Formally, a trajectory is a sequence of states and actions $t = s_0 a_1 s_1 \cdots a_k s_k$ with $s_0 = s_I$. Further, we define the corresponding *label sequence* of t as $t^L \coloneqq l_0 l_1 l_2 \cdots l_k$ where $l_i = L(s_i, a_{i+1}, s_{i+1})$ for each $0 \leq i < k$.

A stationary *policy* $\pi : S \to \Delta(A)$ maps states to probability distributions over the set of actions. In particular, if an agent is in state $s_t \in S$ at time step t and is following policy π, then $\pi(a_t|s_t)$ denotes its probability of taking action $a_t \in A$.

Deterministic Finite Automaton. A *deterministic finite automaton* (DFA) is a finite state machine described using tuple $\mathcal{A} = (V, 2^{\mathcal{P}}, \delta, v_I, F)$ where V is a finite set of states, $2^{\mathcal{P}}$ is the alphabet, $v_I \in V$ is the initial state, $F \subseteq V$ is the set of final states, and $\delta \colon V \times 2^{\mathcal{P}} \mapsto V$ is the deterministic transition function.

A run of a DFA $\mathcal{A}$ on a label sequence $t^L = l_0 l_1 \dots l_k \in (2^{\mathcal{P}})^*$, denoted using $\mathcal{A}\colon v_0 \xrightarrow{t^L} v_{k+1}$, is simply a sequence of states and labels $v_0 l_0 v_1 l_1 \cdots l_k v_{k+1}$, such that $v_0 = v_I$ and for each $0 \leq i \leq k$, $v_{i+1} = \delta(v_i, l_i)$. An accepted run is a run that ends in a final state $v_{k+1} \in F$. Finally, we define the language of $\mathcal{A}$ as $\mathcal{L}(\mathcal{A}) = \{t^L \in (2^{\mathcal{P}})^* \mid t^L \text{ is accepted by } \mathcal{A}\}$.

We define the parallel composition of two DFAs $\mathcal{A}^1 = (V^1, 2^{\mathcal{P}}, \delta^1, v_I^1, F^1)$ and $\mathcal{A}^2 = (V^2, 2^{\mathcal{P}}, \delta^2, v_I^2, F^2)$ to be the cross-product $\mathcal{A}^1 \times \mathcal{A}^2 = (V, 2^{\mathcal{P}}, \delta, v_I, F^2)$, where $V = V^1 \times V^2$, $\delta((s^1, s^2), l_i) = (\delta^1(s^1, l_i), \delta^2(s^2, l_i))$, $v_I = (v_I^1, v_I^2)$, and $F = F^1 \times F^2$. Using such a definition for parallel composition, it is not hard to verify that language $\mathcal{L}(\mathcal{A}^1 \times \mathcal{A}^2)$ is simply $\mathcal{L}(\mathcal{A}^1) \cap \mathcal{L}(\mathcal{A}^2)$.

Task DFA. Following some recent works [1, 19], we rely on so-called *task DFA* $\mathcal{T} = \langle V^{\mathcal{T}}, 2^{\mathcal{P}}, \delta^{\mathcal{T}}, v_I^{\mathcal{T}}, F^{\mathcal{T}} \rangle$ to represent the structure of a non-Markovian reward function. We say a trajectory t has a positive reward if and only if the run of $\mathcal{T}$ on the label sequence t^L, $\mathcal{T}\colon v_0^{\mathcal{T}} \xrightarrow{t^L} v_{k+1}^{\mathcal{T}}$, ends in a final state $v_{k+1}^{\mathcal{T}} \in F^{\mathcal{T}}$.

Linear Temporal Logic. *Linear temporal logic* (over finite traces) (LTL_f) is a logic that expresses temporal properties using temporal modalities. Formally, we define LTL_f formulas—denoted by Greek small letters—inductively as:

- each proposition $p \in \mathcal{P}$ is an LTL_f formula; and
- if ψ and φ are LTL_f formulas, so are $\neg\psi$, $\psi \vee \varphi$, $\mathbf{X}\,\psi$ ("neXt"), and $\psi\,\mathbf{U}\,\varphi$ ("Until").

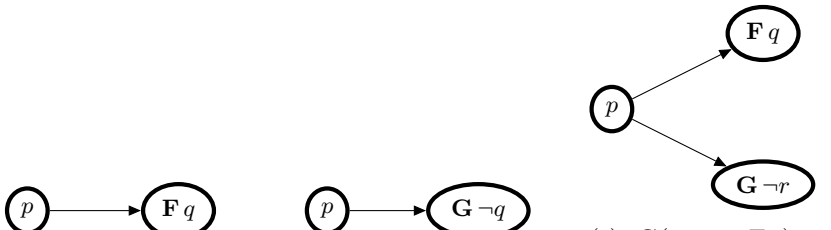
As syntactic sugar, we allow Boolean constants *true* and *false*, and formulas $\psi \wedge \varphi \coloneqq \neg(\neg\psi \vee \neg\varphi)$ and $\psi \rightarrow \varphi \coloneqq \neg\psi \vee \varphi$. Moreover, we additionally allow commonly used temporal formulas $\mathbf{F}\,\psi \coloneqq \mathit{true}\,\mathbf{U}\,\psi$ ("finally") and $\mathbf{G} \coloneqq \neg\,\mathbf{F}\,\neg\varphi$ ("globally").

To interpret LTL_f formulas over (finite) trajectories, we follow the semantics proposed by Giacomo and Vardi [7]. Given a label sequence t^L, we define recursively when an LTL_f formula holds at position i, i.e., $t^L, i \models \varphi$, as follows:

$$
\begin{aligned}
&t^L, i \models p \text{ if and only if } p \in t^L[i] \\
&t^L, i \models \neg\varphi \text{ if and only if } w, i \not\models \varphi \\
&t^L, i \models \varphi \vee \psi \text{ if and only if } t^L, i \models \varphi \text{ or } t^L, i \models \psi \\
&t^L, i \models \mathbf{X}\,\varphi \text{ if and only if } i < |w| \text{ and } t^L, i+1 \models \varphi \\
&t^L, i \models \varphi\,\mathbf{U}\,\psi \text{ if and only if } t^L, j \models \psi \text{ for some} \\
&\qquad\qquad i \leq j \leq |t^L| \text{ and } t^L, i' \models \varphi \text{ for all } i \leq i' < j
\end{aligned}
$$

We say t^L *satisfies* φ if $t^L \models \varphi$, which, in short, is written as $t^L \models \varphi$.

Any LTL_f formula φ can be translated to an equivalent DFA $\mathcal{A}^\varphi$, that is, for any $t^L \in (2^{\mathcal{P}})^*$, $t^L \models \varphi$ if and only if $t^L \in \mathcal{L}(\mathcal{A}^\varphi)$ [7, 31].



(a) $\mathbf{G}(p \rightarrow \mathbf{F}\, q)$ (b) $\mathbf{G}(p \rightarrow \mathbf{G}\, \neg q)$ (c) $\mathbf{G}(p \rightarrow \mathbf{F}\, q) \wedge \mathbf{G}(p \rightarrow \mathbf{G}\, \neg r)$

Fig. 2: Examples of TL-CDs with their corresponding description in LTL

Deterministic causal diagrams. A causal diagram where every edge represents a cause leading to an effect with probability 1 is called a deterministic causal diagram. In a deterministic causal diagram, the occurrence of the cause will always result in the occurrence of the effect.

5 Temporal-Logic-Based Causal Diagrams

We now formalize causality in RL using (deterministic) Causal Diagrams [8], a concept that is widely used in the field of Causal Inference. We here augment Causal Diagrams with temporal logic since we like to express temporally extended relations. We call such Causal Diagrams as Temporal-Logic-based Causal Diagrams or TL-CDs in short. While, in principle, TL-CDs can be conceived for several temporal logics, we consider LTL_f due to its popularity in AI applications [7].

Structurally, for a given set of propositions $\mathcal{P}$, a *Temporal Logic based Causal Diagram* (TL-CD) is a directed acyclic graph $\mathfrak{C}$ where

- each node represents an LTL_f formula over propositions $\mathcal{P}$, and
- each edge ($\rightarrowtail$) represents a causal link between two nodes.

Examples of TL-CDs are illustrated in Figure 2, where, in the causal relation $\psi \rightarrowtail \varphi$, ψ is considered to be the cause and φ to be the effect. The TL-CD in Figure 2a describes that whenever the cause p happens, the effect q eventually (i.e., $\mathbf{F}\, q$) occurs. The TL-CD in Figure 2b describes that whenever the cause p happens, the effect q never (i.e., $\mathbf{G}\, \neg q$) occurs. The TL-CD in Figure 2c describes that whenever the cause p happens, effects q eventually (i.e., $\mathbf{F}\, q$) occurs and r never (i.e., $\mathbf{G}\, \neg r$) occurs.

For a TL-CD to be practically relevant, we must impose that the occurrence of the cause ψ must precede that of the effect φ. To do so, we introduce concepts that track the time of occurrence of an event such as the worst-case satisfaction $w_s(\varphi)$, the worst-case violation $w_v(\varphi)$, the best-case satisfaction $b_s(\varphi)$ and the best-case violation of a formula φ. Intuitively, the worst-case satisfaction $w_s(\varphi)$ (resp., the best-case satisfaction $b_s(\varphi)$) tracks the last (resp., the first) possible

time point that a formula can get satisfied by a trajectory. Likewise, the worst-case violation $w_v(\varphi)$ (resp., the best-case violation $b_v(\varphi)$) tracks the last (resp., the first) possible time point that a formula can get violated by a trajectory. We introduce all the concepts formally in the following definition.

Definition 1. *For an LTL formula φ, we define the worst-case satisfaction time $w_s(\varphi)$, best-case satisfaction time $b_s(\varphi)$, worst-case violation time $w_v(\varphi)$ inductively on the structure of φ as follows:*

$$
\begin{aligned}
b_s(p) =& w_s(p) = b_v(p) = w_v(p) = 0,\\
\neg\varphi :& \begin{cases} b_s(\neg\varphi) = b_v(\varphi),\\ w_s(\neg\varphi) = w_v(\varphi),\\ b_v(\neg\varphi) = b_s(\varphi),\\ w_v(\neg\varphi) = w_s(\varphi); \end{cases}\\
\varphi_1 \wedge \varphi_2 :& \begin{cases} b_s(\varphi_1 \wedge \varphi_2) = \max\{b_s(\varphi_1), b_s(\varphi_2)\},\\ w_s(\varphi_1 \wedge \varphi_2) = \max\{w_s(\varphi_1), w_s(\varphi_2)\},\\ b_v(\varphi_1 \wedge \varphi_2) = \min\{b_v(\varphi_1), b_v(\varphi_2)\},\\ w_v(\varphi_1 \wedge \varphi_2) = \max\{w_v(\varphi_1), w_v(\varphi_2)\}; \end{cases}\\
\varphi_1 \vee \varphi_2 :& \begin{cases} b_s(\varphi_1 \vee \varphi_2) = \min\{b_s(\varphi_1), b_s(\varphi_2)\},\\ w_s(\varphi_1 \vee \varphi_2) = \max\{w_s(\varphi_1), w_s(\varphi_2)\},\\ b_v(\varphi_1 \vee \varphi_2) = \max\{b_v(\varphi_1), b_v(\varphi_2)\},\\ w_v(\varphi_1 \vee \varphi_2) = \max\{w_v(\varphi_1), w_v(\varphi_2)\}; \end{cases}\\
\mathbf{G}\varphi :& \begin{cases} b_s(\mathbf{G}\varphi) = w_s(\mathbf{G}\varphi) = w_v(\mathbf{G}\varphi) = \infty,\\ b_v(\mathbf{G}\varphi) = b_v(\varphi); \end{cases}\\
\mathbf{F}\varphi :& \begin{cases} b_s(\mathbf{F}\varphi) = b_s(\varphi),\\ w_s(\mathbf{F}\varphi) = w_v(\mathbf{F}\varphi) = b_v(\mathbf{F}\varphi) = \infty; \end{cases}\\
\mathbf{X}\varphi :& \begin{cases} b_s(\mathbf{X}\varphi) = b_s(\varphi) + 1,\\ w_s(\mathbf{X}\varphi) = w_s(\varphi) + 1,\\ w_v(\mathbf{X}\varphi) = w_v(\varphi) + 1,\\ b_v(\mathbf{X}\varphi) = b_v(\varphi) + 1; \end{cases}\\
\varphi_1\mathbf{U}\varphi_2 :& \begin{cases} b_s(\varphi_1\mathbf{U}\varphi_2) = b_s(\varphi_2),\\ w_s(\varphi_1\mathbf{U}\varphi_2) = w_v(\varphi_1\mathbf{U}\varphi_2) = \infty,\\ b_v(\varphi_1\mathbf{U}\varphi_2) = b_v(\varphi_1). \end{cases}
\end{aligned}
$$

For each causal relation $\psi \rightarrowtriangle \varphi$ in a causal diagram $\mathfrak{C}$, we impose the constraints that $w_s(\psi) \leq \min\{b_s(\varphi), b_v(\varphi)\}$ and $w_v(\psi) \leq \min\{b_s(\varphi), b_v(\varphi)\}$. Such a constraint is designed to make sure that the cause ψ, even in the worst-time scenario, occurs before the event φ, in the best-time scenario. Based on the con-

straint, we rule out causal relations in which the cause occurs after the effect such as $\mathbf{X}\, p \twoheadrightarrow q$, where $w_s(\mathbf{X}\, p) = 1$ is greater than $b_s(q) = 0$.

To express the meaning of TL-CD in formal logic, we turn to its description in LTL. A causal relation $\psi \twoheadrightarrow \varphi$ can be described using the LTL_f formula $\mathbf{G}(\psi \rightarrow \varphi)$, which expresses that whenever ψ occurs, φ should also occur. Further, an entire TL-CD $\mathfrak{C}$ can be described using the LTL_f formula $\varphi^{\mathfrak{C}} \coloneqq \bigwedge_{(\psi \twoheadrightarrow \varphi)} \mathbf{G}(\psi \rightarrow \varphi)$ which is simply the conjunction of the LTL_f formulas corresponding to each causal relation in $\mathfrak{C}$.

Based on its description in LTL_f $\varphi^{\mathfrak{C}}$, we can now define when a trajectory π satisfies a TL-CD $\mathfrak{C}$. Precisely, t satisfies $\mathfrak{C}$ if and only if its label sequence t^L satisfies $\varphi^{\mathfrak{C}}$.

In the subsequent sections, we also rely on a representation of a TL-CD as a deterministic finite automaton (DFA). In particular, for a TL-CD $\mathfrak{C}$, we can construct a DFA $\mathcal{C}^{\mathfrak{C}} = \langle V^{\mathcal{C}}, 2^{\mathcal{P}}, \delta^{\mathcal{C}}, v_I^{\mathcal{C}}, F^{\mathcal{C}} \rangle$ from its description in LTL_f $\varphi^{\mathfrak{C}}$. We call such a DFA a *causal DFA*. When the TL-CD is clear from the context, we simply represent a causal DFA as $\mathcal{C}$, dropping its superscript.

In the motivating example (Section 2), the TL-CD $\mathfrak{C}$ pictured in Figure 1b translates into the LTL_f formula $\varphi^{\mathfrak{C}} = \mathbf{G}(p \rightarrow \mathbf{X}\, g) \wedge \mathbf{G}(s \rightarrow \mathbf{G} \neg \mathbf{X}\, b)$, which is equivalent to the causal DFA $\mathcal{C}$ pictured in Figure 1c.

6 Reinforcement Learning with Causal Diagrams

We now aim to utilize the information provided in a Temporal-Logic-based Causal Diagram (TL-CD) to enhance the process of reinforcement learning in a non-Markovian setting. However, in our setting, we assume a TL-CD to be a ground truth about the causal relations in the underlying environment. As a result, we must ensure that a TL-CD is compatible with a labeled MDP.

Intuitively, a TL-CD $\mathfrak{C}$ is compatible with a labeled MDP $\mathcal{M}$ if all possible trajectories of $\mathcal{M}$ respect (i.e., do not violate) the TL-CD $\mathfrak{C}$. To define compatibility formally, we rely on the cross-product $\overline{\mathcal{M}} \times \mathcal{C}$, where $\overline{\mathcal{M}}$ is a (non-deterministic) finite state machine representation of $\mathcal{M}$ with states S, alphabet $2^{\mathcal{P}}$, transition $\delta(s, l) = \{s' \in S \mid L(s, a, s') = l \text{ for some } a \in A\}$, initial state s_I and final states S, and $\mathcal{C}$ is the causal DFA.

Formally, we say that a TL-CD $\mathfrak{C}$ is *compatible* with an MDP $\mathcal{M}$ if from any reachable state $(s, q) \in \overline{\mathcal{M}} \times \mathcal{C}$, one can always reach a state $(s', q') \in \overline{\mathcal{M}} \times \mathcal{C}$ where q' is a final state in the causal DFA $\mathcal{C}$. The above formal definition ensures that any trajectory of $\mathcal{M}$ can be continued to satisfy the causal relations defined by $\mathfrak{C}$.

We are now ready to state the central problem of the paper.

Problem 1 (Non-Markovian Reinforcement learning with Causal Diagrams). Let $\mathcal{M}$ be a labeled MDP, $\mathcal{T}$ be a task DFA, and $\mathfrak{C}$ be a Temporal Logic based Causal Diagram (TL-CD) such that $\mathfrak{C}$ is compatible with $\mathcal{M}$. Given $\mathcal{M}$, $\mathcal{T}$ and $\mathfrak{C}$, learn a policy that achieves a maximal reward in the environment.

We view TL-CDs as a concise representation of the causal knowledge in an environment. In our next subsection, we develop an algorithm that exploits this causal knowledge to alleviate the issues of extensive interaction in an online RL setting.

6.1 Q-learning with early stopping

Our RL algorithm is an adaptation of QRM [11], which is a Q-learning algorithm [23] that is typically used when rewards are specified as finite state machines. On a high level, QRM explores the product space $\mathcal{M} \times \mathcal{T}$ in many episodes in the search for an optimal policy. We modify QRM by stopping its exploration early based on the causal knowledge from a TL-CD $\mathfrak{C}$. Before we describe the algorithm in detail, we must introduce some concepts that aid the early stopping.

For early stopping to work, the learning agent must keep track of whether a trajectory can lead to a reward. We do this by keeping track of the current configuration in the synchronized run of a trajectory on the product $\mathcal{T} \times \mathcal{C}$ of the task DFA and the causal DFA. We here identify two particular configurations that can be useful for early stopping: *causally accepting* and *causally rejecting*. Intuitively, a trajectory reaches a causally accepting configuration if all continuations of the trajectory from the current configuration that satisfy the TL-CD $\mathfrak{C}$ receive a reward of 1 (or a positive reward). On the other hand, a trajectory reaches a causally rejecting configuration if all continuations of the trajectory rom the current configuration that satisfy the TL-CD $\mathfrak{C}$, do not receive a reward.

We formalize the notion of causally accepting configurations and causal rejecting configurations in the following two definitions. We use the terminology $\mathcal{A}_q$ to describe a DFA that is structurally identical to DFA $\mathcal{A}$, except that its initial state is q.

Definition 2 (Causally accepting). *We say $(v^{\mathcal{T}}, v^{\mathcal{C}}) \in V^{\mathcal{T}} \times V^{\mathcal{C}}$ is causally accepting if for each $t^L \in (2^{\mathcal{P}})^*$ for which the run $\mathcal{C} : v^{\mathcal{C}} \xrightarrow{t^L} v_f^{\mathcal{C}}$ ends in some final state $v_f^{\mathcal{C}} \in F^{\mathcal{C}}$, the run $\mathcal{T} : v^{\mathcal{T}} \xrightarrow{t^L} v_f^{\mathcal{T}}$ must also end in some final state $v_f^{\mathcal{T}} \in F^{\mathcal{T}}$. Equivalently, we say that $(v^{\mathcal{T}}, v^{\mathcal{C}}) \in V^{\mathcal{T}} \times V^{\mathcal{C}}$ is causally accepting if $\mathcal{L}(\mathcal{C}_{v^{\mathcal{C}}}) \subseteq \mathcal{L}(\mathcal{T}_{v^{\mathcal{T}}})$.*

Definition 3 (Causally rejecting). *We say $(v^{\mathcal{T}}, v^{\mathcal{C}}) \in V^{\mathcal{T}} \times V^{\mathcal{C}}$ is causally rejecting if for each $t^L \in (2^{\mathcal{P}})^*$ for which the run $\mathcal{C} : v^{\mathcal{C}} \xrightarrow{t^L} v_f^{\mathcal{C}}$ ends in some final state $v_f^{\mathcal{C}} \in F^{\mathcal{C}}$, the run $\mathcal{T} : v^{\mathcal{T}} \xrightarrow{t^L} v_f^{\mathcal{T}}$ must not end in any final state in $F^{\mathcal{T}}$. Equivalently, we say that $(v^{\mathcal{T}}, v^{\mathcal{C}}) \in V^{\mathcal{T}} \times V^{\mathcal{C}}$ is causally rejecting if $\mathcal{L}(\mathcal{C}_{v^{\mathcal{C}}}) \cap \mathcal{L}(\mathcal{T}_{v^{\mathcal{T}}}) = \emptyset$.*

Remark 1. A configuration $(v^{\mathcal{T}}, v^{\mathcal{C}})$ may be neither causally accepting nor causally rejecting.

To illustrate these concepts, we consider the motivating example introduced in Section 2, where $\mathcal{T}$ and $\mathcal{C}$ are depicted in Figures 1a and 1c. The initial state $(v_0^{\mathcal{T}}, v_0^{\mathcal{C}})$ is neither causally accepting nor causally rejecting. If the agent encounters a label p, it reaches $(v_1^{\mathcal{T}}, v_1^{\mathcal{C}})$, which is causally accepting since the only reachable configurations where $\mathcal{C}$ is accepting $\{(v_3^{\mathcal{T}}, v_0^{\mathcal{C}}), (v_3^{\mathcal{T}}, v_2^{\mathcal{C}})\}$ are accepting for $\mathcal{T}$. If the agent encounters a label s instead, it reaches $(v_2^{\mathcal{T}}, v_2^{\mathcal{C}})$, which is causally rejecting since the only reachable configurations where $\mathcal{C}$ is accepting $\{(v_2^{\mathcal{T}}, v_2^{\mathcal{C}})\}$ are rejecting for $\mathcal{T}$.

Algorithm 1: causally accepting/rejecting state detection

Input: Task DFA $\mathcal{T}$, Causal DFA $\mathcal{C}$, a pair of states $(v^{\mathcal{T}}, v^{\mathcal{C}}) \in V^{\mathcal{T}} \times V^{\mathcal{C}}$.
$C^{\mathcal{T}} \leftarrow \emptyset$ `// set of reachable "causal states" of` $\mathcal{T}$
$\mathfrak{P} \leftarrow \mathcal{T} \times \mathcal{C}$ `// the parallel composition of` $\mathcal{T}$ `and` $\mathcal{C}$
foreach *state* $(v_r^{\mathcal{T}}, v_r^{\mathcal{C}})$ *of* $\mathfrak{P}$ *reachable from* $(v^{\mathcal{T}}, v^{\mathcal{C}})$ **do**
    if $v_r^{\mathcal{C}} \in F^{\mathcal{C}}$ **then**
        $C^{\mathcal{T}} \leftarrow C^{\mathcal{T}} \cup \{v_r^{\mathcal{T}}\}$
// $(v^{\mathcal{T}}, v^{\mathcal{C}})$ `is causally accepting if` $C^{\mathcal{T}} \subseteq F^{\mathcal{T}}$
// $(v^{\mathcal{T}}, v^{\mathcal{C}})$ `is causally rejecting if` $C^{\mathcal{T}} \cap F^{\mathcal{T}} = \emptyset$
return $C^{\mathcal{T}}$

We now present the pseudo-code of the algorithm used for detecting the causally accepting and causally rejecting configurations in Algorithm 1. Intuitively, the algorithm relies on a breadth-first search on the cross-product DFA $\mathcal{T} \times \mathcal{C}$ of the task DFA and the causal DFA.

Algorithm 2: Q-learning with TL-CD

Input: Labeled MDP $\mathcal{M}$, Task DFA $\mathcal{T}$, Causal diagram $\mathfrak{C}$.
Convert $\mathfrak{C}$ to causal DFA $\mathcal{C}$
Detect causally accepting and rejecting states of $\mathcal{T} \times \mathcal{C}$
foreach *training episode* **do**
    run QTLCD_episode

We now expand on our adaptation of the QRM algorithm. The pseudo-code of the algorithm is sketched in Algorithm 2. In the algorithm, we first compute the set of causally accepting or causally rejecting configurations as described in Algorithm 1. Next, like a typical Q-learning algorithm, we perform explorations of the environment in several episodes to estimate the Q-values of the state-action pairs. However, during an episode, we additionally keep track of the configuration of the product $\mathcal{T} \times \mathcal{C}$. If during the episode, we encounter a causally accepting

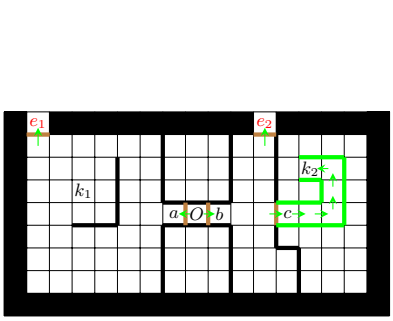
or causally rejecting configuration, we terminate the episode and update the Q-values accordingly.

Algorithm 3: QTLCD_episode

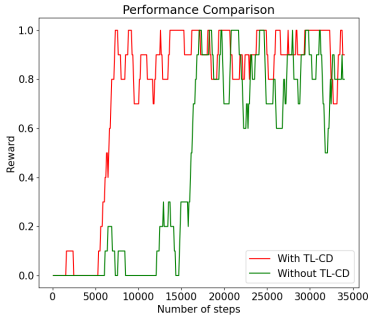
Hyperparameter: Q-learning parameters, episode length *eplength*.
Input: labeled MDP $\mathcal{M}$, task DFA $\mathcal{T}$, causal DFA $\mathcal{C}$, learning rate α.
Input: a set of q-functions $Q = \{q^{v^\mathcal{T}} | v^\mathcal{T} \in V^\mathcal{T}\}$
Output: the updated set of q-functions Q
$s \leftarrow s_I; v^\mathcal{T} \leftarrow v_I^\mathcal{T}; v^\mathcal{C} \leftarrow v_I^\mathcal{C}$ `// initialise states`
$R \leftarrow 0$ `// initialise cumulative reward`
for $0 \leq t <$ *eplength* **do**
  $a \leftarrow$ GetEpsilonGreedyAction$(q^{v^\mathcal{T}}, s)$ `// get action from policy`
  $s' \leftarrow ExecuteAction(p(s, a))$ `// based on distribution` $p(s, a)$
  $v^{\mathcal{T}\prime} \leftarrow \delta^\mathcal{T}(v^\mathcal{T}, L(s, a, s'))$ `// synchronize` $\mathcal{T}$
  $v^{\mathcal{C}\prime} \leftarrow \delta^\mathcal{C}(v^\mathcal{C}, L(s, a, s'))$ `// synchronize` $\mathcal{C}$
  $R' \leftarrow \mathbb{1}_{F^\mathcal{T}}(v^{\mathcal{T}\prime})$ `// compute cumulative reward`
  `// override reward based on causal analysis:`
  if $(v^{\mathcal{T}\prime}, v^{\mathcal{C}\prime})$ *causally accepting* **then** $R' \leftarrow 1$
  if $(v^{\mathcal{T}\prime}, v^{\mathcal{C}\prime})$ *causally rejecting* **then** $R' \leftarrow 0$
  update $q^{v^\mathcal{T}}(s, a)$ using reward $r = R' - R$ `// Bellman update`
  if $v^{\mathcal{T}\prime} \in F^\mathcal{T}$ **then return** Q `// end of episode`
  if $(v^{\mathcal{T}\prime}, v^{\mathcal{C}\prime})$ *causally accepting or rejecting* **then return** Q `// interrupt episode early`
  $s \leftarrow s'; v^\mathcal{T} \leftarrow v^{\mathcal{T}\prime}; v^\mathcal{C} \leftarrow v^{\mathcal{C}\prime}; R \leftarrow R'$
return Q

The Q-learning with TL-CD algorithms consists of a loop of several episodes. The pseudo-code of one episode is sketched in Algorithm 3. The instant reward r is computed such that the cumulative reward R is 1 if and only if the Task DFA is accepting. The cumulative reward is then overridden if it is possible to predict the future cumulative reward, based on if the current configuration is causally accepting or causally rejecting. If the reward could be predicted, the episode is interrupted right after updating the q-functions, using that predicted reward. Note that the notion causally accepting and rejecting configurations is defined on unbounded episodes, and might predict a different reward than if the episode were to time out.

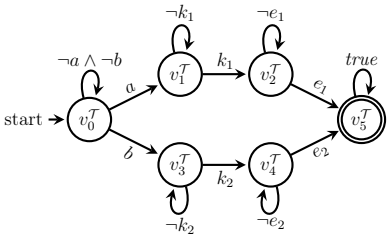
The above algorithm follows the exact steps of the QRM algorithm and thus, inherits all its advantages, including termination and optimality. The only notable difference is the early stopping based on the causally accepting and causally rejecting states. However, when these configurations are reached, based on their definition, all continuations are guaranteed to return positive and no reward, respectively. Thus, early stopping helps to determine the future reward



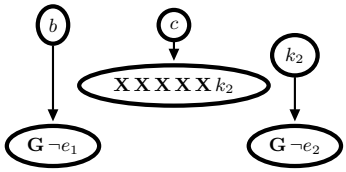
(a) Map of the environment



(b) Performance comparison



(c) Task DFA $\mathcal{T}$



(d) TL-CD $\mathfrak{C}$

Fig. 3: Case study I: small office world.

and update the estimates of Q-value earlier. We next demonstrate the advantages of the algorithm experimentally.

7 Case Studies

In this section, we implement the proposed Q-learning with TL-CD (`QTLCD`) algorithm in comparison with a baseline algorithm in three different case studies.

In each case study, we compare the performance of the following two algorithms:

- Q-learning with TL-CD (`QTLCD`): the proposed algorithm, including early stopping of the episodes when a causally accepting/rejecting state is reached
- Q-learning with Reward Machines (`QRM`): the algorithm from [11], with the same MDP and RM but no causal diagram.

7.1 Case Study I: Small Office World Domain

We consider a small officeworld scenario in a 17×9 grid. The agent's objective is to first reach the location of either one key k_1 or k_2 and then exit the grid by reaching either e_1 or e_2. The agent navigates on the grid with walls, keys, and one-way doors. The set of actions is $A = \{S, N, E, W\}$. The action S, N, E, W

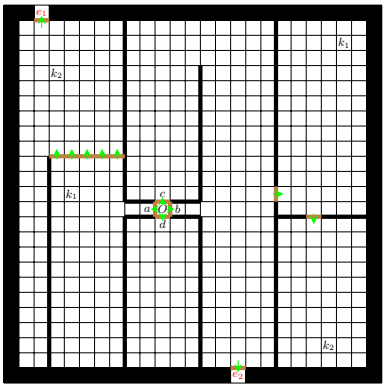
correspond to moving in the four cardinal directions. The one-way doors are shown in Figure 3a with green arrows. We specify the complex task of the RL agent in a maze as a *deterministic finite automaton (DFA)* $\mathcal{T}$ (see Fig. 3), as both event sequences $a - k_1 - e_1$ (open door a, pick up key k_1, exit at e_1) and $b - k_2 - e_2$ (open door b, pick up key k_2, exit at e_2) lead to completion of the task of exiting the maze (receiving reward 1). The agent starts at position o. If TL-CD in Figure 3d is true, then the RL agent should never go along the sequence $b - k_2 - e_2$ as $k_2 \dashrightarrow \mathbf{G}\, \neg e_2$ means if the agent picks up key k_2 then it can never exit at e_2 (as c is a one-way door, so the agent can never get outside Room 3 once it enters c to pick up key k_2).

Results: Figure 3b (d) presents the performance comparison of the RL agent with TL-CD and without TL-CD. It shows that the accumulated reward of the RL agent can converge to its optimal value 3 times faster if the agent knows the TL-CD and learns never to open door b.

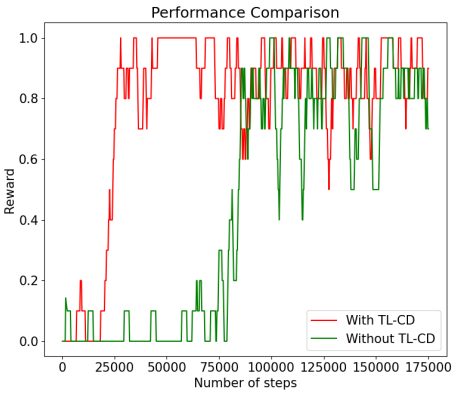
7.2 Case Study II: Large Office World Domain

We consider a large office world scenario in a 25×25 grid. The objective of the agent is to exit the grid from e_1 or e_2 after visiting the location of both keys k_1 and k_2 in a given sequence, here the agent first has to reach the location of k_1 and then the location of k_2. The set of actions is $A = \{S, N, E, W\}$. The action S, N, E, W correspond to moving in the four cardinal directions. The one-way doors are shown in Figure 4a with green arrows. The motivation behind this example is to observe the effect of increasing causally rejecting states on RL agents' performance. The task DFA and the TL-CD are depicted in Section 7.2.

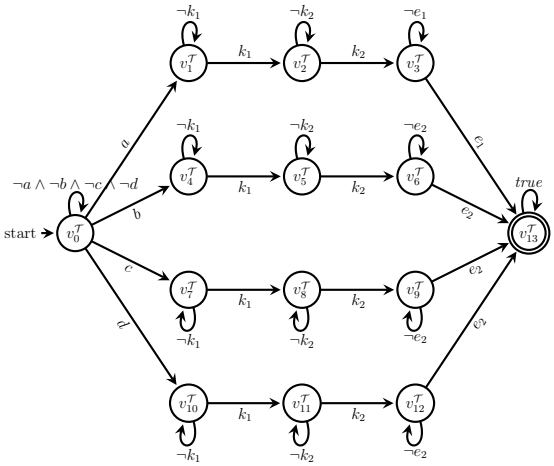
Results : Figure 4b presents the performance comparison of the RL agent in a large office world scenario with TL-CD (`QTLCD`) and without TL-CD (`QRM`). It shows that the RL agent can converge to its optimal value 5 times faster if the agent knows the TL-CD.



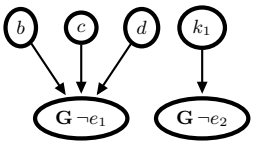
(a) Environment Map



(b) Performance comparison



(c) Task DFA $\mathcal{T}$



(d) TL-CD $\mathfrak{C}$

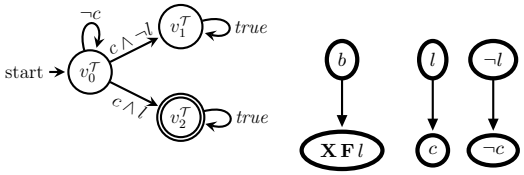
Fig. 4: Case study II: large office world environment.

7.3 Case Study III: Crossroad Domain

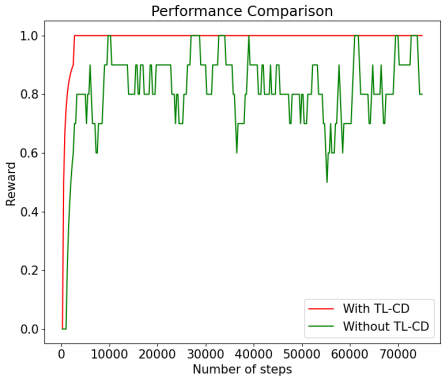
This experiment is inspired by the real-world example of crossing the road at a traffic signal. The agent's objective is to reach the other side of the road. The agent navigates on a grid with walls, crossroad, button, and light signal. The agent starts from a random location in the grid. The set of actions is $A = \{S, N, E, W, PressButton, Wait\}$. The action $PressButton$ presses the button at the crossroad to indicate it wants to cross the road. After pressing the button, at some later time, the pedestrian crossing light will be turned ON. The action $Wait$ will let the agent stay at a location. The actions S, N, E, W correspond to moving in the four cardinal directions.

To simplify the problem, we make some assumptions: the agent starts from a fixed location in the left half (one side) of the grid. After pressing the button at the crossroad, the crossing light will turn ON N steps later, where N is a random variable following a geometric distribution of success probability 0.05. Thus, the underlying MDP has three observable variables: x, y, the discrete coordinates of the agent on the grid, and t, a Boolean flag that indicates that the button has been pressed and that the light is bound to happen (note that a geometric law is memoryless and does not require extra variables). We specify the task as to reach the location of the crossroad where the button is located, press the button, and cross the road only when the light signal is ON. We define two labels for the task: e for successfully crossing the road when the light is ON, and f for crossing the road when the light is not ON.

We consider the causal LTL specification $\mathbf{G}(b \rightarrow \mathbf{F}\,\mathbf{X}\,l) \wedge \mathbf{G}(l \leftrightarrow c)$, where the first part of the conjunction represents the knowledge that the pedestrian light has to turn ON some time later, and the second part represents the policy of the agent, because we suppose that the agent already knows to cross if and only if the light is on. Under these conditions, pressing the button leads to a causally accepting state.



(a) Task DFA $\mathcal{T}$ (b) TL-CD $\mathfrak{C}$



(c) Performance comparison

Fig. 5: Case study III: crossword domain.

8 Conclusions and discussions

This paper introduces the Temporal-Logic-based Causal Diagram (TL-CD) in reinforcement learning (RL) to address the limitations of traditional RL algorithms that use finite state machines to represent temporally extended goals. By capturing the temporal causal relationships between different properties of the environment, our TL-CD-based RL algorithm requires significantly less exploration of the environment and can identify expected rewards early during exploration. Through a series of case studies, we demonstrate the effectiveness of our algorithm in achieving optimal policies with faster convergence than traditional RL algorithms.

In the future, we plan to explore the applicability of TL-CDs in other RL settings, such as continuous control tasks and multi-agent environments. Additionally, we aim to investigate the scalability of TL-CDs in large-scale environments and the impact of noise and uncertainty on the performance of the algorithm. Another direction for future research is to investigate the combination of TL-CDs with other techniques, such as meta-learning and deep reinforcement learning, to further improve the performance of RL algorithms in achieving temporally extended goals.

References

1. Abate, A., Almulla, Y., Fox, J., Hyland, D., Wooldridge, M.J.: Learning task automata for reinforcement learning using hidden markov models. CoRR **abs/2208.11838** (2022)
2. Aksaray, D., Jones, A., Kong, Z., Schwager, M., Belta, C.: Q-learning for robust satisfaction of signal temporal logic specifications. In: IEEE CDC'16. pp. 6565–6570 (Dec 2016). https://doi.org/10.1109/CDC.2016.7799279
3. Alshiekh, M., Bloem, R., Ehlers, R., Könighofer, B., Niekum, S., Topcu, U.: Safe reinforcement learning via shielding. In: AAAI'18 (2018)
4. Bareinboim, E., Forney, A., Pearl, J.: Bandits with unobserved confounders: A causal approach. In: Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 28. Curran Associates, Inc. (2015), https://proceedings.neurips.cc/paper/2015/file/795c7a7a5ec6b460ec00c5841019b9e9-Paper.pdf
5. Forney, A., Pearl, J., Bareinboim, E.: Counterfactual data-fusion for online reinforcement learners. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 70, pp. 1156–1164. PMLR (06–11 Aug 2017), https://proceedings.mlr.press/v70/forney17a.html
6. Fu, J., Topcu, U.: Probably approximately correct MDP learning and control with temporal logic constraints. Robotics: Science and Systems **abs/1404.7073** (2014)
7. Giacomo, G.D., Vardi, M.Y.: Linear temporal logic and linear dynamic logic on finite traces. In: IJCAI. pp. 854–860. IJCAI/AAAI (2013)
8. Greenland, S., Pearl, J.: Causal diagrams. In: International Encyclopedia of Statistical Science, pp. 208–216. Springer (2011)
9. Howard, R., Matheson, J.: Influence diagrams. Decision Analysis **2**, 127–143 (09 2005). https://doi.org/10.1287/deca.1050.0020
10. Huang, B., Feng, F., Lu, C., Magliacane, S., Zhang, K.: AdaRL: What, where, and how to adapt in transfer reinforcement learning. ArXiv **abs/2107.02729** (2021)
11. Icarte, R.T., Klassen, T.Q., Valenzano, R.A., McIlraith, S.A.: Using reward machines for high-level task specification and decomposition in reinforcement learning. In: ICML. Proceedings of Machine Learning Research, vol. 80, pp. 2112–2121. PMLR (2018)
12. Koller, D., Milch, B.: Multi-agent influence diagrams for representing and solving games. Games and Economic Behavior **45**(1), 181–221 (2003). https://doi.org/https://doi.org/10.1016/S0899-8256(02)00544-4, https://www.sciencedirect.com/science/article/pii/S0899825602005444, first World Congress of the Game Theory Society
13. Lattimore, F., Lattimore, T., Reid, M.D.: Causal bandits: Learning good interventions via causal inference (2016). https://doi.org/10.48550/ARXIV.1606.03203, https://arxiv.org/abs/1606.03203
14. Lee, S., Bareinboim, E.: Structural causal bandits with non-manipulable variables. Proceedings of the AAAI Conference on Artificial Intelligence **33**(01), 4164–4172 (Jul 2019). https://doi.org/10.1609/aaai.v33i01.33014164, https://ojs.aaai.org/index.php/AAAI/article/view/4320
15. Lee, S., Bareinboim, E.: Characterizing optimal mixed policies: Where to intervene and what to observe. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) Advances in Neural Information Processing Systems. vol. 33, pp. 8565–8576. Curran Associates, Inc. (2020), https://proceedings.neurips.cc/paper/2020/file/61a10e6abb1149ad9d08f303267f9bc4-Paper.pdf

16. Lee, S., Correa, J.D., Bareinboim, E.: General identifiability with arbitrary surrogate experiments. In: Adams, R.P., Gogate, V. (eds.) Proceedings of The 35th Uncertainty in Artificial Intelligence Conference. Proceedings of Machine Learning Research, vol. 115, pp. 389–398. PMLR (22–25 Jul 2020), https://proceedings.mlr.press/v115/lee20b.html
17. Li, X., Vasile, C.I., Belta, C.: Reinforcement learning with temporal logic rewards. In: Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst. pp. 3834–3839 (Sept 2017). https://doi.org/10.1109/IROS.2017.8206234
18. Lu, C., Huang, B., Wang, K., Hernández-Lobato, J.M., Zhang, K., Schölkopf, B.: Sample-efficient reinforcement learning via counterfactual-based data augmentation (2020). https://doi.org/10.48550/ARXIV.2012.09092, https://arxiv.org/abs/2012.09092
19. Memarian, F., Xu, Z., Wu, B., Wen, M., Topcu, U.: Active task-inference-guided deep inverse reinforcement learning. In: CDC. pp. 1932–1938. IEEE (2020)
20. Neider, D., Gaglione, J.R., Gavran, I., Topcu, U., Wu, B., Xu, Z.: Advice-guided reinforcement learning in a non-markovian environment. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 9073–9080 (2021)
21. Pitis, S., Creager, E., Garg, A.: Counterfactual data augmentation using locally factored dynamics. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. NIPS'20, Curran Associates Inc., Red Hook, NY, USA (2020)
22. Spirtes, P.: Introduction to causal inference. Journal of Machine Learning Research **11**(54), 1643–1662 (2010), http://jmlr.org/papers/v11/spirtes10a.html
23. Sutton, R.S., Barto, A.G.: Reinforcement learning - an introduction. Adaptive computation and machine learning, MIT Press (1998)
24. Tennenholtz, G., Mannor, S., Shalit, U.: Off-policy evaluation in partially observable environments. ArXiv **abs/1909.03739** (2019)
25. Wen, M., Papusha, I., Topcu, U.: Learning from demonstrations with high-level side information. In: Proc. IJCAI'17. pp. 3055–3061 (2017). https://doi.org/10.24963/ijcai.2017/426, https://doi.org/10.24963/ijcai.2017/426
26. Xu, Z., Gavran, I., Ahmad, Y., Majumdar, R., Neider, D., Topcu, U., Wu, B.: Joint inference of reward machines and policies for reinforcement learning. In: Proc. International Conference on Automated Planning and Scheduling (ICAPS), Special Track on Planning and Learning (2020)
27. Xu, Z., Topcu, U.: Transfer of temporal logic formulas in reinforcement learning. In: Proc. IJCAI'2019. pp. 4010–4018 (7 2019). https://doi.org/10.24963/ijcai.2019/557, https://doi.org/10.24963/ijcai.2019/557
28. Xu, Z., Wu, B., Ojha, A., Neider, D., Topcu, U.: Active finite reward automaton inference and reinforcement learning using queries and counterexamples. In: Holzinger, A., Kieseberg, P., Tjoa, A.M., Weippl, E. (eds.) Machine Learning and Knowledge Extraction. pp. 115–135. Springer International Publishing, Cham (2021)
29. Zhang, A., Lipton, Z.C., Pineda, L., Azizzadenesheli, K., Anandkumar, A., Itti, L., Pineau, J., Furlanello, T.: Learning causal state representations of partially observable environments. ArXiv **abs/1906.10437** (2019)
30. Zhang, J., Bareinboim, E.: Transfer learning in multi-armed bandits: A causal approach. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence. p. 1340–1346. IJCAI'17, AAAI Press (2017)
31. Zhu, S., Tabajara, L.M., Li, J., Pu, G., Vardi, M.Y.: Symbolic ltlf synthesis. In: IJCAI. pp. 1362–1369. ijcai.org (2017)