Proceedings of the Thirty-Third International Conference on Automated Planning and Scheduling (ICAPS 2023)

Reinforcement Learning for Omega-Regular Specifications on Continuous-Time MDP

Amin Falah[1], **Shibashis Guha**[2], **Ashutosh Trivedi**[1]
[1] University of Colorado Boulder
[2] Tata Institute of Fundamental Research
amin.falah@colorado.edu, shibashis.guha@tifr.res.in, ashutosh.trivedi@colorado.edu

Abstract

Continuous-time Markov decision processes (CTMDPs) are canonical models to express sequential decision-making under dense-time and stochastic environments. When the stochastic evolution of the environment is only available via sampling, model-free reinforcement learning (RL) is the algorithm-of-choice to compute optimal decision sequence. RL, on the other hand, requires the learning objective to be encoded as scalar reward signals. Since doing such translations manually is both tedious and error-prone, a number of techniques have been proposed to translate high-level objectives (expressed in logic or automata formalism) to scalar rewards for discrete-time Markov decision processes. Unfortunately, no automatic translation exists for CTMDPs.

We consider CTMDP environments against the learning objectives expressed as omega-regular languages. Omega-regular languages generalize regular languages to infinite-horizon specifications and can express properties given in popular linear-time logic LTL. To accommodate the dense-time nature of CTMDPs, we consider two different semantics of omega-regular objectives: 1) *satisfaction semantics* where the goal of the learner is to maximize the probability of spending positive time in the good states, and 2) *expectation semantics* where the goal of the learner is to optimize the long-run expected average time spent in the "good states" of the automaton. We present an approach enabling correct translation to scalar reward signals that can be readily used by off-the-shelf RL algorithms for CTMDPs. We demonstrate the effectiveness of the proposed algorithms by evaluating it on some popular CTMDP benchmarks with omega-regular objectives.

Introduction

Reinforcement learning (RL) is a sequential optimization approach where a decision maker learns to optimally resolve a sequence of choices based on feedback received from the environment. This feedback often takes the form of rewards and punishments with strength proportional to the fitness of the decisions taken by the agent as judged by the environment towards some higher-level learning objectives. *This paper develops convergent RL algorithms for continuous-time Markov decision processes (CTMDP) against learning requirements expressed in ω-regular languages.*

Need for Reward Translation. Due to a combination of factors—including the success of deep neural networks (Goodfellow, Bengio, and Courville 2016) and a heavy intellectual and monetary investment from the industry and the academia (Mnih et al. 2015; Silver et al. 2016)—RL has emerged as a leading human-AI collaborative design paradigm where the key role of the human designers reduces to designing the appropriate scalar reward signals, while the RL algorithm creates an optimal schedule driven by the reward signal. Unfortunately, then, the de-facto communication between the human designers and the RL algorithms is quite rigid: it forces the human programmers to think in the language suitable for the learning agents and not in a way that comes naturally to humans: declarative or imperative languages. To meet this challenge, a recent trend is to enable logic (Sadigh et al. 2014; Camacho et al. 2019; Li, Vasile, and Belta 2017) and automatic structures (Hahn et al. 2019; Icarte et al. 2018, 2022) to express learning intent in RL. The common thread among these approaches is to encode the specification as an automaton based reward structure and derive scalar rewards with every discrete interaction with the environment. However, when the problem domain is continuous-time, aforementioned approaches are not applicable as they support the discrete-time semantics modeled as finite-state Markov decision processes (MDP or DTMDP for emphasis).

This paper aims to enable the use of RL in unknown CTMDPs against high-level specifications expressed as ω-automata (Vardi and Wolper 1986; Baier and Katoen 2008).

Continuous-Time Reinforcement Learning. Semi-MDPs (Baykal-Gürsoy 2011) model environments where the interaction between the decision maker and the environment may occur at any dense time point. CTMDPs (Guo and Hernández-Lerma 2009) are subclasses of semi-Markov decision processes where the exact time and the resolution of the next state is governed by an exponential distribution with a *rate* parameter that is dependent on the current state and the action chosen. The classical RL algorithms for DTMDPs have been elegantly generalized to CTMDPs for both discounted (Bradtke and Duff 1994) and average (Das et al. 1999) objectives. We employ the Q-learning algorithm for CTMDPs (Bradtke and Duff 1994) to compute optimal schedules for ω-regular learning objectives.

The ω-Regular Objectives. Finite automata on infinite words—or ω-automata—may be equipped with a variety of equally expressive infinitary acceptance conditions (e.g., deterministic Rabin and nondeterministic Büchi) with well-understood succinctness and complexity trade-offs. From their first application in solving Church's synthesis problem (Thomas 2009) to becoming the *lingua franca* in expressing specifications of safety-critical systems (Baier and Katoen 2008), ω-automata are a key part of the computational backbone to automated verification and synthesis. Linear temporal logic (LTL) (Baier and Katoen 2008) is a popular declarative language to express properties of infinite sequences. Specifications expressed using ω-automata form a strict superset of specifications expressed as LTL formulas. Given an LTL formula, one can effectively construct an ω-automaton (Vardi and Wolper 1986). For this reason, we focus on ω-automata based specifications.

The expanding role of RL in safety-critical systems has prompted the use of ω-automata in expressing learning objectives due to improved expressiveness and interpretability over scalar rewards. In this work, we use nondeterministic Büchi automata to express ω-regular specifications.

Continuous-Time in Büchi Automata. Büchi automata are finitary structures accepting infinite sequences of letters that visit a distinguished set of good (accepting) states infinitely often. For scheduling problems over stochastic systems modeled as DTMDPs, the optimal schedules can be specified via schedules that maximize the measure of accepted system behaviors. While for discrete-time system the naturalness of such discrete infinitary visitation semantics is well-established, for continuous-time systems it is imperative that the acceptance criterion must heed to the actual time spent in such good states. Two distinct interpretations of good dense-time behavior are natural: While the focus of the *satisfaction semantics* is on maximizing the measure of behaviors that visit good states infinitely often, the *expectation semantics* focuses on maximizing the long-run expected time spent in good states. We develop RL algorithms for CTMDPs with Büchi specifications under both semantics.

A recent work (Oura and Ushio 2022) studies an alternative objective for semi-MDPs against multi-objective specifications composed of an ω-regular objectives (satisfaction semantics) and a risk objective (expected risk). The key distinction between Oura and Ushio's approach and ours (vis-à-vis the satisfaction semantics) is that the former is based on bounded synthesis paradigm that requires a bound parameter on co-Büchi states visitation and thus reduces the specification to a safety objective (where reward translation is straightforward). In contrast, our approach does not require any bound from the practitioner and is capable of handling general ω-regular objectives. Moreover, the expectation semantics has not been explored in any existing literature.

Contributions. Our key contributions are as follows:

1. We present a novel (expectation) semantics for Büchi automata to capture time-critical properties for CTMDPs.
2. We present procedures to translate Büchi automata with satisfaction and expectation semantics to reward machines (Icarte et al. 2018) in a form that enables application of the off-the-shelf CTMDP RL algorithms. We show that one needs distinct reward mechanisms for these two semantics, and we establish the correctness and effectiveness of these reward translations.
3. We present an experimental evaluation to demonstrate the effectiveness of the proposed approach.

A full version of the paper with the detailed proofs can be found in (Falah, Guha, and Trivedi 2023).

Satisfaction Vs. Expectation Semantics

We motivate for the satisfaction and the expectation semantics by presenting a simple example. The CTMDP shown in Figure 1, adapted from (Hahn et al. 2019), represents four zones (s_0 to s_3 in the figure) on the Mars surface. Suppose that a mission to Mars arrives in Zone 0 (a known, safe territory) and is expected to explore the terrain in a safe fashion, gather and transmit information, and stay alive to maximize the return on the mission. For simplicity, assume that Zone 1 (purple) models a crevasse harmful to the safe operations, while zones 2 and 3 are central to exploration mission and are analogous in their information contents.

Given the unknown uncertainty of the terrain of Mars, the system is modeled as a CTMDP with associated uncertainty on the time of various actions where the exit rate of action a from Zone (state) 0 is denoted by $\lambda(0, a)$. In other words, when selected an action a in a state s, the probability of spending t time units in s before taking a is given by the cumulative distribution function $1 - e^{-\lambda(0,a)t}$. Each transition have a rate associated with it and it determines the probability of taking that transition. Assume that the action b from Zone 0 goes to Zone 2 with rate r (high probability) and to Zone 1 with rate $(\lambda(0, b) - r)$ (low probability). The mission objective is to avoid Zone 1 (purple zone) while infinitely often visiting the Zone 2 or 3 (the green zones). It can be captured in LTL (Baier and Katoen 2008) as:

$$\varphi = (\mathsf{G}\,\neg \mathsf{p}) \wedge (\mathsf{G}(\mathsf{F}\,\mathsf{g}))$$

specifying that across the infinite horizon always (i.e. at every step expressed as temporal modality, G) avoid the purple region ($\neg b$), and always eventually (i.e. at some time in the future expressed as temporal modality, F) reach the green region, i.e. $(\mathsf{G}(\mathsf{F}\,g))$. The $GF\phi$ modality is often referred as *infinitely often* ϕ. LTL combines these temporal operators using the standard propositional logic connectives such as: and ($\wedge$), or ($\vee$), not ($\neg$), and implication ($\rightarrow$).

This declarative specification can also be expressed using the Büchi automaton shown in Figure 1 (center) where the double circled states (here, q_1) denote accepting states. The Büchi automaton can be used as a monitor to check the behavior of the learner over the environment. For our example, it is visualized by taking the synchronous product of the CTMDP with the automaton shown in Figure 1 (right).

For the satisfaction semantics on the product CTMDP, our goal is to maximize the probability that every infinite horizon behavior visits the accepting state infinitely often, while for the expectation semantics the goal is to maximize the expected time the system dwells in the accepting state.

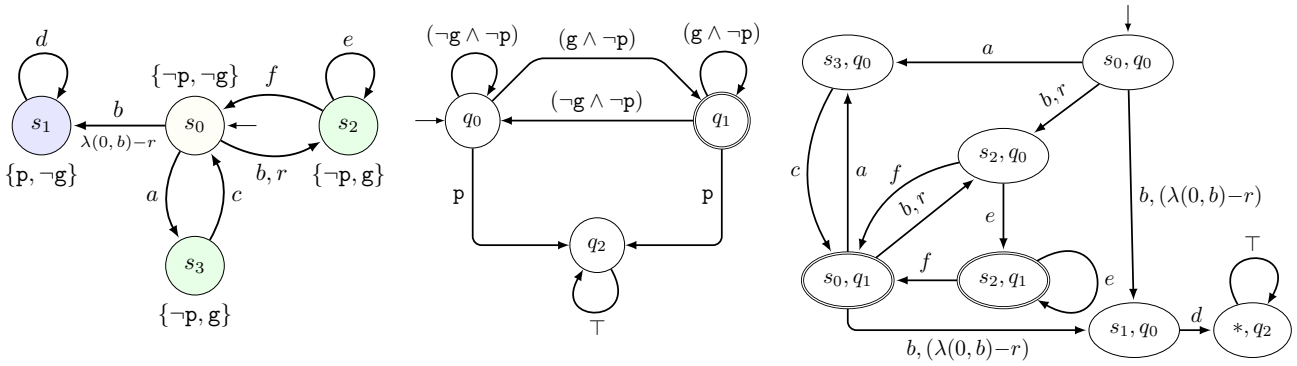

Figure 1: The mars surveillance example where a CTMDP (left) can be in four different states where states s_0 has the label $\{\neg\mathsf{p}, \neg\mathsf{g}\}$, state s_2 and s_3 have label $\{\neg\mathsf{p}, \mathsf{g}\}$, and state s_1 have the label $\{\mathsf{p}, \neg\mathsf{g}\}$. The rates of each transition (if not 1) is written in the figure. A deterministic Büchi automata for the ω-regular objective $\varphi = (\mathsf{G}\,\neg\mathsf{p}) \wedge (\mathsf{G}\,\mathsf{F}\,\mathsf{g})$ (center). Product CTMDP (right) where each zone has two components for denoting the CTMDP and the Büchi automaton parts. All the zones whose second component is q_2 is combined as one. The exit rate of an action from a zone (s, q_i) in the product CTMDP is same as the exit rate of the action from s in the original CTMDP.

- **Satisfaction Objective.** Consider the case where we have one Mars rover in this mission. Hence, our goal is to maximize the probability of visiting green zones infinitely often while avoiding the purple zone (the satisfaction semantics). In this case, the optimal schedule is to choose actions a and c indefinitely, i.e. the schedule $(a \to c)^\omega$, that satisfies the objective with probability 1. Note that action b is always sub-optimal irrespective of the probability to move to the purple zone.
- **Expectation Objective.** Consider an alternative setting where we have a fleet of drones (we are okay in losing some drones as long as we maximize the mission objective) that needs to be sent to the surveillance of zone 2 or 3. Suppose that due to unforeseeable circumstances the mission may cease operation any time, and hence the goal is to maximize total expected time spent in the green zones (2 and 3). The schedule $(a \to c)^\omega$ is not optimal anymore as it may dwell a considerable amount in the Zone 0. On the other hand any drone that chooses b in Zone 0 risks moving to Zone 1 with a small probability. As our goal is to maximize the expected time spent in the green zone over a large group of drones, the expectation semantics captures this intent and the optimal schedule is to start with action b.

Preliminaries

We write $\mathbb{N}, \mathbb{Q}$ and $\mathbb{Q}_{\geq 0}$ for the sets of natural numbers, rational numbers, and non-negative rational numbers, respectively. For a natural number $n \in \mathbb{N}$, we denote by $[n]$ the set $\{1, \ldots, n\}$. Given a finite set A, a (rational) *probability distribution* over A is a function $p\colon A \to [0,1] \cap \mathbb{Q}$ such that $\sum_{a \in A} p(a) = 1$ and $\mathsf{Supp}(p) = \{a \in A \mid p(a) > 0\}$ is the *support* of p. We denote the set of probability distributions on A by $\mathcal{D}(A)$.

Continuous-Time MDPs. A (discrete-time) *Markov decision process (MDP)* is a tuple of the form $\mathcal{M} = (\mathsf{S}, s_0, \mathsf{Act}, \mathbb{T})$, where S is a finite set of *states*, $s_0 \in \mathsf{S}$ is the *initial* state, Act is a finite set of *actions*, and $\mathbb{T} : \mathsf{S} \times \mathsf{Act} \to \mathcal{D}(\mathsf{S})$ is a *transition function*. Let $\mathsf{Act}(s)$ be the set of actions enabled in the state $s \in S$. An MDP is called a Markov chain if for every $s \in S$, the set $\mathsf{Act}(s)$ is singleton.

A *continuous-time* MDP (CTMDP) is a tuple of the form $\mathcal{M} = (\mathsf{S}, s_0, \mathsf{Act}, \mathsf{R})$, where $\mathsf{R} : \mathsf{S} \times \mathsf{Act} \times \mathsf{S} \to \mathbb{R}_{\geq 0}$ is a *transition rate function*, while the rest of the parameters are same as that of an MDP. For $s \in \mathsf{S}$ and $a \in \mathsf{Act}(s)$, we define $\lambda(s,a) = \sum_{s'} \mathsf{R}(s,a,s') > 0$ to be the *exit rate* of a in s. We define a *probability matrix*, $P_{\mathcal{M}}$, where

$$P_{\mathcal{M}}(s,a,s') = \begin{cases} \frac{\mathsf{R}(s,a,s')}{\lambda(s,a)} & \text{if } \lambda(s,a) > 0 \\ 0 & \text{otherwise} \end{cases}$$

When $\mathcal{M}$ is clear from the context, we simply denote $P_{\mathcal{M}}$ by P. The residence time for action a in s is exponentially distributed with mean $\lambda(s,a)$. For a given state s and an action a, the probability of spending t time units in s before taking the action is given by the cumulative distribution function $F(t|s,a) = 1 - e^{-\lambda(s,a)t}$ of the exponential distribution. The probability of a transition from state s to s' on an action a in t time units is $p_a(s,s',t) = P(s,a,s') \cdot F(t|s,a)$. A CTMDP is called a continuous-time Markov chain (CTMC) if for every state $s \in S$, the set $\mathsf{Act}(s)$ is singleton.

Uniformization. A *uniform* CTMDP has a constant exit rate C for all state-action pairs i.e, $\lambda(s,a) = C$ for all states $s \in \mathsf{S}$ and actions $a \in \mathsf{Act}(s)$. The procedure of converting a non-uniform CTMDP into a uniform one is known as *uniformization*. Consider a non-uniform CTMDP $\mathcal{M}$. Let $C \in \mathbb{R}_{\geq 0}$ be such that $C \geq \lambda(s,a)$ for all $(s,a) \in \mathsf{S} \times \mathsf{Act}$. We obtain a uniform CTMDP $\mathcal{M}_C$ by changing the rates to R':

$$\mathsf{R}'(s,a,s') = \begin{cases} \mathsf{R}(s,a,s') & \text{if } s \neq s' \\ \mathsf{R}(s,a,s') + C - \lambda(s,a) & \text{if } s = s' \end{cases}$$

For every action $a \in \mathsf{Act}(s)$ from each state s in the new CTMDP we have a self loop if $\lambda(s,a) < C$. A uniformized

CTMDP has a constant transition rate C for all actions and because of this, the mean interval time between any two successive actions is constant.

Schedules. An infinite run of the CTMDP is an ω-word $(s_1,(t_1,a_1),s_2,(t_2,a_2),\ldots) \in \mathsf{S} \times ((\mathbb{R}_{\geq 0} \times \mathsf{Act}) \times \mathsf{S})^\omega$ where $s_i \in \mathsf{S}$, $a_i \in \mathsf{Act}(s_i)$ and t_i is the time spent on state s_i. A finite run is of the form $(s_1,t_1,a_1,\ldots,t_{n-1},a_{n-1},s_n)$ for some $n \in \mathbb{N}$. The set of infinite and the set of finite runs in $\mathcal{M}$ are denoted by $Runs^{\mathcal{M}}$ and $FRuns^{\mathcal{M}}$ respectively. Similarly, $Runs^{\mathcal{M}}(s)$ and $FRuns^{\mathcal{M}}(s)$ represents the finite and infinite runs starting from state s. For $r \in FRuns^{\mathcal{M}}$, we denote by $last(r)$ the last state in the run r.

We use a *schedule* to resolve non-determinism in a CTMDP. A schedule is a function $\sigma : FRuns^{\mathcal{M}} \to \mathcal{D}(\mathsf{Act})$, where $\mathcal{D}(\mathsf{Act})$ is a probability distribution on the set of enabled actions. For a finite run $r \in FRuns^{\mathcal{M}}$, a schedule gives a probability distribution over all actions enabled in $last(r)$. A schedule is *deterministic* if $\mathcal{D}(\mathsf{Act})$ is Dirac, i.e, a single action is chosen in the distribution, otherwise it is *randomized*. Further, a schedule σ is stationary if for all r,$r' \in FRuns^{\mathcal{M}}$ with $last(r) = last(r')$, we have that $\sigma(r) = \sigma(r')$. A *pure* schedule is a deterministic stationary schedule. Let $\Sigma_{\mathcal{M}}$ be the set of all schedules.

A CTMDP $\mathcal{M}$ under a schedule σ acts as a continuous time Markov chain (CTMC) which is denoted by $\mathcal{M}^{[\sigma]}$. The set of infinite and the set of finite runs in $\mathcal{M}^{[\sigma]}$ are denoted by $Runs_\sigma^{\mathcal{M}}$ and $FRuns_\sigma^{\mathcal{M}}$ respectively. The behavior of a CTMDP $\mathcal{M}$ under a schedule σ and starting state $s \in S$ is defined on a probability space $(Runs_\sigma^{\mathcal{M}}(s), FRuns_\sigma^{\mathcal{M}}(s), \Pr_\sigma^{\mathcal{M}}(s))$ over the set of infinite runs of σ with starting state s. Given a random variable $f : Runs_\sigma^{\mathcal{M}} \to \mathbb{R}$, we denote by $\mathbb{E}_\sigma^{\mathcal{M}}(s)\{f\}$ the expectation of f over the runs of $\mathcal{M}^{[\sigma]}$. For $n \geq 1$, we write X_n, Y_n, D_n, and T_n for the random variables corresponding to the n-th state, action, time-delay in the n-th state, and time-stamp (time spent up to the n-th state). We let $D_0 = T_0 = 0$.

Rewardful CTMDPs. A rewardful CTMDP $(\mathcal{M}, rew)$ is a CTMDP and a reward function $rew : \mathsf{S} \cup (\mathsf{S} \times \mathsf{Act}) \to \mathbb{R}_{\geq 0}$ which assigns a *reward-rate* to each state and a scalar reward to each state-action pair. Thus spending t time-units in $s \in \mathsf{S}$ gives $rew(s) \cdot t$ of (state-delay) reward and choosing a from s gives $rew(s,a)$ (action) reward.

Continuous time discounting is done with respect to a discount parameter $\alpha{>}0$ where one unit of reward obtained at time t in the future gets a value of $e^{-\alpha t}$. Formally, the expected discounted reward for an arbitrary schedule σ from a state s is given by:

$$\mathsf{DR}^{\mathcal{M}[\sigma]}(\alpha)(s) = \mathbb{E}_\sigma^{\mathcal{M}}(s)\Big[\sum_{n=1}^{\infty} e^{-\alpha T_{n-1}}\Big(rew(X_n,Y_n) + \int_{T_{n-1}}^{T_n} e^{-\alpha(t-T_{n-1})} rew(X_n)dt\Big)\Big].$$

Here, we multiply the expected reward obtained at the $n{-}$th state with $e^{-\alpha T_{n-1}}$ as per the continuous time discounting. The initial term in the parenthesis corresponds to the reward obtained from state X_n by picking action Y_n (action reward) and the second term corresponds to the state-delay reward i.e reward obtained with respect to the reward-rate $rew(X_n)$ which is discounted over the time $(t - T_{n-1})$.

The expected average reward from s under σ is given by:

$$\mathsf{AR}^{\mathcal{M}[\sigma]}(s) = \liminf_{N\to\infty} \mathbb{E}_\sigma^{\mathcal{M}}(s)\Big[\frac{1}{T_N} \cdot \Big(\sum_{n=1}^{N} rew(X_n,Y_n) + \int_{T_{n-1}}^{T_n} rew(X_n)dt\Big)\Big],$$

where the first and second term corresponds to the action and state-delay reward respectively. Recall that T_N is the total time spent upto the n-th state. Consider an objective $\mathcal{O} \in \{\mathsf{DR}, \mathsf{AR}\}$. The expected reward obtained by schedule σ on $s \in \mathsf{S}$ is denoted by $\mathcal{O}^{\mathcal{M}[\sigma]}(s)$. A schedule σ^* is optimal for $\mathcal{O}$ if $\mathcal{O}^{\mathcal{M}[\sigma^*]}(s) = sup_{\sigma\in\Sigma_{\mathcal{M}}} \mathcal{O}^{\mathcal{M}[\sigma]}(s)$ for all $s \in \mathsf{S}$.

For a given CTMDP $\mathcal{M}$, one can compute the optimal schedule for the discounted-sum objective or the expected average by using policy iteration, value iteration or linear programming (Feinberg and Shwartz 2002; Puterman 2014) on the uniformized CTMDP $\mathcal{M}_C$. When the CTMDP is unknown (unknown rates and states), an optimal schedule can be computed via reinforcement learning.

Reinforcement Learning (RL). RL allows us to obtain an optimal schedule by repeatedly interacting with the environment and thereby observing a reward. A *training episode* is a finite sequence of states, actions and rewards which terminates on certain specified conditions like when the number of samples drawn is greater than some threshold. The RL obtains information about rates and rewards of the CTMDP model by running several training episodes. Broadly, there are two categories of RL, model-based and model-free. We focus on space efficient model-free RL algorithms as they compute optimal schedule without constructing the state transition system (Strehl et al. 2006).

One of the most successful model-free learning algorithm for DTMDPs is the Q-learning algorithm (Watkins and Dayan 1992). It aims at learning (near) optimal schedules in a (partially unknown) MDP for the discounted sum objective. Bradtke and Duff (Bradtke and Duff 1994) introduced the Q-learning algorithm for CTMDPs. We give here a brief description of Q-learning algorithm for CTMDPs.

For a given discount parameter $\alpha{>}0$, the one-step expected discounted reward for an action a from state s is given by $\rho(s,a) = rew(s,a) + \frac{rew(s)}{\alpha+\lambda(s,a)}$ (Puterman 2014, Eq. 11. 5. 3). The Q-function for a state s and an action a under schedule σ, denoted $\mathcal{Q}_\sigma(s,a)$, is defined as

$$\rho(s,a) + \frac{\lambda(s,a)}{\lambda(s,a)+\alpha} \sum_{s'\in\mathsf{S}} P(s,a,s') \cdot \mathcal{Q}_\sigma(s',\sigma(s')).$$

It gives the total expected discounted reward obtained by taking action a from s, and following σ afterwards. The optimal Q-function, denoted $\mathcal{Q}^*$ is given by,

$$\rho(s,a) + \frac{\lambda(s,a)}{\lambda(s,a)+\alpha} \sum_{s'\in\mathsf{S}} P(s,a,s') \cdot \max_{a'\in\mathsf{Act}} \mathcal{Q}^*(s',\alpha').$$

Q-learning uses stochastic approximation (Sutton and Barto 2018) to estimate the $\mathcal{Q}^*$ function. When a transition from state s to s' on an action a with delay τ is observed, the $\mathcal{Q}_f$ estimates are updated as (Bradtke and Duff 1994, Eq 12):

$$\mathcal{Q}_f^{(k+1)}(s,a) := (1-\beta_k)\mathcal{Q}_f^{(k)}(s,a) + \beta_k\Big(r(s,a,s') + e^{-\alpha\tau}\max_{a'}\mathcal{Q}_f^{(k)}(s',a')\Big),$$

where $r(s,a,s')$ is the sampled reward from state s to s', the sampled transition time is τ, and β_k is the learning rate. The RL algorithm samples through states and updates the Q-function iteratively. The optimal schedule is generated after completion of some number of episodes by taking the action that gives the highest Q-value from each state.

We focus on how to *automatically obtain reward mechanisms for ω-regular objectives for CTMDPs* so that off-the-shelf RL algorithms can learn an optimal schedule.

Problem Statement

Omega-regular Objectives. An ω-regular objective is defined by a nondeterministic Büchi automaton $\mathcal{A} = (\Sigma, Q, q_0, \delta, F)$ where Σ is a finite *alphabet*, Q is a finite set of *states*, $q_0 \in Q$ is an *initial state*, $\delta : Q\times\Sigma \to 2^Q$ is a *transition function* and $F \subseteq Q$ is the set of *accepting states*. A Büchi automaton is deterministic, if $\delta(q,a)$ is singleton for all $(q,a) \in Q\times\Sigma$. We define the extended transition function $\hat{\delta} : Q \times \Sigma^* \to 2^Q$, derived from δ, as $\hat{\delta}(q,\varepsilon) = \{q\}$ and $\hat{\delta}(q,ax) = \cup_{q'\in\delta(q,a)}\hat{\delta}(q',x)$, for $q \in Q$ and $ax \in \Sigma\Sigma^*$.

A *run* r of $\mathcal{A}$ is an infinite sequence $(r_0, w_0, r_1, w_1, \ldots)$ where $r_0 = q_0$, $r_i \in Q$, $w_i \in \Sigma$ and $r_{i+1} \in \delta(r_i, w_i)$ for all $i \in \mathbb{N}$. The word of a run $r = (r_0, w_0, r_1, w_1, \ldots)$ is $\mathsf{L}(r) = (w_0 w_1 \cdots)$. Let the set of runs of $\mathcal{A}$ be $\mathcal{R}_\mathcal{A}$. We say that a run $r \in \mathcal{R}_\mathcal{A}$ is accepting if there exists a $q_f \in F$ such that q_f occurs infinitely often in r. An ω-word $w = (w_0 w_1 \cdots)$ is accepted by $\mathcal{A}$ if there exists an accepting run $r_w = (r_0, w_0, r_1, w_1, \ldots)$ of $\mathcal{A}$. The language of the automaton $\mathcal{A}$, denoted $\mathcal{L}(\mathcal{A})$ is the set of all words that is accepted by the automaton.

CTMDPs and Omega-regular Objectives. In order to express the properties of a CTMDP $\mathcal{M}$ using a Büchi automaton, we introduce the notion of a labelled CTMDP. A *labelled* CTMDP is a triple $(\mathcal{M}, \mathbf{AP}, L)$ where $\mathcal{M}$ is a CTMDP, $\mathbf{AP}$ is a set of atomic propositions, and $L : \mathsf{S} \to 2^{\mathbf{AP}}$ is a labelling function. Let $\mathcal{A} = (2^{\mathbf{AP}}, Q, q_0, \delta, F)$ be a Büchi automaton expressing the learning objectives of $\mathcal{M}$.

Recall that for a CTMDP $\mathcal{M}$ under a schedule σ we write X_n, Y_n, D_n, and T_n for the random variables corresponding to the n-th state, action, time-delay at the n-th state, and time-stamp (total time spent up to the n-th state). We introduce the random variable F_n to indicate if the sequence of observations of the CTMDP leads to an accepting state on $\mathcal{A}$ in n-steps, i.e., $F_n = [\hat{\delta}(L(X_0)\cdot L(X_1)\cdots L(X_n)) \cap F]$.

For a CTMDP $(\mathcal{M}, \mathbf{AP}, L)$ and automaton $\mathcal{A} = (2^{\mathbf{AP}}, Q, q_0, \delta, F)$, we study the following problems:

1. **Satisfaction Semantics.** Compute a schedule of $\mathcal{M}$ that maximizes the probability of visiting accepting states F of $\mathcal{A}$ infinitely often. We define the satisfaction probability of a schedule σ from starting state s as:

$$\mathsf{PSem}_\mathcal{A}^\mathcal{M}(s,\sigma) = \Pr_\sigma^\mathcal{M}(s)\left\{\forall_i \exists_{j\geq i} F_j\right\}.$$

   Intuitively, it describes the probability of runs from state s under σ in the CTMDP such that the corresponding run in $\mathcal{A}$ visits the accepting states infinitely often. The optimal satisfaction probability $\mathsf{PSem}_\mathcal{A}^\mathcal{M}(s)$ for $\mathcal{A}$ is defined as $\sup_{\sigma\in\Sigma_\mathcal{M}} \mathsf{PSem}_\mathcal{A}^\mathcal{M}(s,\sigma)$, and we say that a schedule $\sigma \in \Sigma_\mathcal{M}$ is optimal for $\mathcal{A}$ if $\mathsf{PSem}_\mathcal{A}^\mathcal{M}(s,\sigma) = \mathsf{PSem}_\mathcal{A}^\mathcal{M}(s)$ for all $s \in \mathsf{S}$.
2. **Expectation Semantics.** Compute a schedule of $\mathcal{M}$ that maximizes the long-run expected average time spent in the accepting states of $\mathcal{A}$. We define the expected satisfaction time of a schedule σ from starting state s as:

$$\mathsf{ESem}_\mathcal{A}^\mathcal{M}(s,\sigma) = \mathbb{E}_\sigma^\mathcal{M}(s)\left\{\liminf_{n\to\infty}\frac{\sum_{i=1}^n F_i\cdot D_i}{T_n}\right\}.$$

   The optimal expected satisfaction time $\mathsf{ESem}_\mathcal{A}^\mathcal{M}(s)$ for specification $\mathcal{A}$ is defined as $\sup_{\sigma\in\Sigma_\mathcal{M}} \mathsf{ESem}_\mathcal{A}^\mathcal{M}(s,\sigma)$, and we say that $\sigma \in \Sigma_\mathcal{M}$ is an optimal expectation maximisation schedule for $\mathcal{A}$ if $\mathsf{ESem}_\mathcal{A}^\mathcal{M}(s,\sigma) = \mathsf{ESem}_\mathcal{A}^\mathcal{M}(s)$.

Product Construction. Given a *labelled* CTMDP $(\mathcal{M}, \mathbf{AP}, L)$ where $\mathbf{AP}$ is a set of atomic propositions, and $L : \mathsf{S} \to 2^{\mathbf{AP}}$ is a labelling function, and a Büchi automaton $\mathcal{A} = (2^{\mathbf{AP}}, Q, q_0, \delta, F)$, the product CTMDP is defined as $\mathcal{M}\times\mathcal{A} = ((\mathsf{S}\times Q), (s_0,q_0), \mathsf{Act}, \mathsf{R}^\times, F^\times)$ where the rates are $\mathsf{R}^\times : (\mathsf{S}\times Q)\times\mathsf{Act}\times(\mathsf{S}\times Q) \to \mathbb{R}_{\geq 0}$ such that $\mathsf{R}^\times((s,q),a,(s',q')) = \mathsf{R}(s,a,s')$ if $\mathsf{R}(s,a,s') > 0$ and $\delta(q,L(s)) = \{q'\}$. If F is the set of accepting states in $\mathcal{A}$, then the accepting condition is a set $F^\times$ of states where $(s,q) \in F^\times$ iff $q \in F$.

Good-for-CTMDP Automata. From the definition of both the semantics, it is clear that the optimal schedule requires some memory to monitor the run in the Büchi automaton (see Example 2 in Appendix D in (Falah, Guha, and Trivedi 2023)). For the right kind of Büchi automata (Hahn et al. 2020), the amount of memory required can be equal to the size of the automata. A key construction to compute these schedules is the product construction, where the CTMDP and the automaton are combined together as a CTMDP with accepting states governed by the accepting states of the Büchi automata. On the other hand, not every Büchi automaton can be used for this construction. The class of Büchi automata where the semantic value of satisfaction of the property on the MDP equals to the corresponding problems on the product structure, are called good-for-MDP (GFM) automata (Hahn et al. 2020).

If a Büchi automaton is GFM, then one can show via uniformization that it is also good-for-CTMDPs. (See Appendix C in (Falah, Guha, and Trivedi 2023) for a formal definition of good-for-CTMDPs.) There exist several syntactic characterizations of good-for-MDP automata including suitable limit-deterministic Büchi automata (SLDBA) (Sickert et al. 2016) and slim automata (Hahn et al. 2020). Moreover, every LTL specification can be effectively converted

into a GFM Büchi automata and there exist tools (OWL and Spot) to convert LTL objectives to good-for-CTMDP automaton. Hence, in this paper, w.l.o.g., we assume that ω-regular objectives are given as good-for-CTMDP automata.

Problem Definition. Given a CTMDP $\mathcal{M}$ with unknown transition structure and rates, and an ω-regular objective ϕ given as a good-for-CTMDP Büchi automata $\mathcal{A}$, we are interested in the following reward translation problem for the satisfaction semantics and for the expectation semantics.

Problem 1 (Reward Translation Scheme)**.** *Design a reward scheme for $\mathcal{A}$ such that any off-the-shelf RL algorithm optimizing the discounted reward in CTMDPs converges to an optimal schedule for satisfaction (expectation) semantics.*

In Section we provide a solution for the satisfaction semantics, while in Section we sketch a solution for this problem for the expectation semantics. We reduce these problems to average reward maximization for CTMDPs. Since average-reward RL algorithms for CTMDPs and MDPs require strong assumptions on the structure (such as communicating MDPs) (Sutton and Barto 2018), we solve the average-reward RL problem by reducing it to a discounted-reward problem using the following result.

Theorem 2. *For every CTMDP $\mathcal{M}$, there exists a pure schedule σ^* and a threshold $0 \leq \gamma^{\mathcal{M}}_{\alpha_0} < 1$ such that for every discount-rate function γ_α, where $\gamma_\alpha(s,a) \geq \gamma^{\mathcal{M}}_{\alpha_0}$ for every valid state-action pair (s,a), the schedule σ^* is an optimal schedule maximising the expected discounted reward. Moreover, σ^* also maximizes the expected average reward.*

This schedule σ^* is known as a Blackwell optimal schedule. We show that we need different reward translation schemes for the two semantics.

RL for Satisfaction Semantics

We reduce the problem of satisfaction semantics of an ω-regular objective in a CTMDP to an expected average reward objective. Using Blackwell optimality result stated in Theorem 2, we further reduce this to an expected discounted reward objective which allows us to use off-the-shelf RL for CTMDP for learning schedules for ω-regular objectives.

To find a schedule satisfying an ω-regular objective in a CTMDP, we identify the accepting end-components where an accepting end-component (De Alfaro 1998) is a sub-MDP that is closed under probabilistic transitions and contains an accepting state. It is known (De Alfaro 1998) that as an end-component C of an CTMDP is entered, there is a schedule that visits every state-action pair in C with probability 1 and stays in C forever. Hence, a schedule that maximizes the probability of satisfaction of a given ω-regular objective maximizes the probability of reaching the accepting end-components. The CTMDP in Figure 2(top) is itself an accepting end-component since the state q_0 is accepting.

We further reduce the problem to an average reward problem as described below and then specify a reward function such that the schedule maximising the expected average reward maximizes the probability of satisfying the objective.

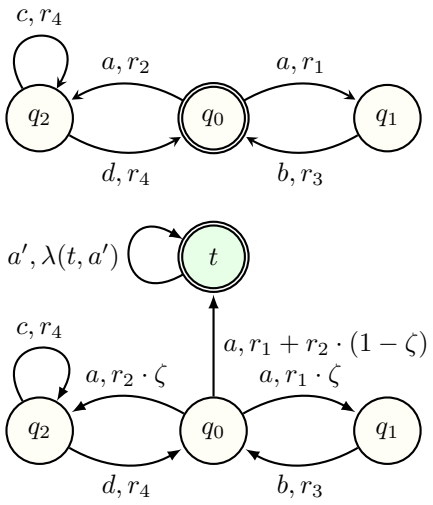

Figure 2: A product CTMDP ($\mathcal{M} \times \mathcal{A}$) (top) and its corresponding augmented product CTMDP $\mathcal{M}^\zeta$ (bottom).

Reduction to Average Reward. Before describing our RL algorithm for unknown CTMDP, we first describe the reduction when an input CTMDP is fully known to explain the intuition behind our algorithm. Consider a CTMDP $\mathcal{M}$, a GFM $\mathcal{A}$, and let $\mathcal{M} \times \mathcal{A}$ denote the product CTMDP. For our reduction, we define a constant $\zeta \in (0,1)$ and an augmented product CTMDP, denoted by $\mathcal{M}^\zeta$. The CTMDP $\mathcal{M}^\zeta$ is constructed from $\mathcal{M} \times \mathcal{A}$ by adding a new sink state t with a self loop labelled by an action a' and with rate $\lambda(t,a') > 0$, and making it the only accepting state in $\mathcal{M}^\zeta$. Further, in $\mathcal{M}^\zeta$, the rates of each outgoing transition from an accepting state in $\mathcal{M} \times \mathcal{A}$ is multiplied by ζ. Also, for each action a from an accepting state s in $\mathcal{M} \times \mathcal{A}$, in $\mathcal{M}^\zeta$ we add a new transition to the sink state t with rate $\lambda(s,a) \cdot (1-\zeta)$ where $\lambda(s,a)$ is the exit rate of the state-action pair (s,a) in $\mathcal{M} \times \mathcal{A}$. Figure 2 shows an example of this construction. Note that in the figure, q_0 is the only accepting state in the product CTMDP. There are two outgoing transitions from q_0 on action a to q_1 and q_2 with rates r_1 and r_2 respectively, and hence $\lambda(q_0,a) = r_1 + r_2$. We then add a transition from q_0 to t with rate $(r_1 + r_2) \cdot (1-\zeta)$.

With a slight abuse of notation, if σ is a schedule in the augmented CTMDP $\mathcal{M}^\zeta$, then we also denote by σ a schedule in $\mathcal{M} \times \mathcal{A}$ obtained by removing t from the domain of σ. Thus fix a schedule σ in both $\mathcal{M}^\zeta$ and in $\mathcal{M} \times \mathcal{A}$. Note that for every state in an accepting end-component, the probability of reaching the sink t in $\mathcal{M}^\zeta$ is 1. Similarly, for every state in a rejecting end-component, the probability of reaching t in $\mathcal{M}^\zeta$ is 0. The probability of reaching t in $\mathcal{M}^\zeta$ under σ overapproximates the probability of reaching the accepting end-components in $\mathcal{M} \times \mathcal{A}$ under σ. The difference in the two probabilities occurs since in $\mathcal{M}^\zeta$, from the transient accepting states, with probability $1-\zeta$, one can reach the sink t. This approximation error tends to 0 as ζ tends to 1. We define a reward function in $\mathcal{M}^\zeta$ such that a schedule maximising the expected average reward in $\mathcal{M}^\zeta$ maximizes the probability of satisfying the ω regular objective in $\mathcal{M} \times \mathcal{A}$.

Reward Function. The reward function provides a reward of 1 per time unit for staying in the accepting sink t, while the reward is 0 otherwise, i.e.

$$rew(s) = \begin{cases} 1 & \text{if } s = t \\ 0 & \text{otherwise} \end{cases}$$

As there is only a single action a' from state t in $\mathcal{M}^\zeta$ which is a self loop, we can conclude that any schedule that maximizes the probability of reaching t also maximizes the expected average reward in $\mathcal{M}^\zeta$. Following the discussion above, for high values of ζ, the schedule also maximizes the probability of satisfying the ω-regular objective in $\mathcal{M} \times \mathcal{A}$. We thus have the following.

Theorem 3. *There exists a threshold $\zeta' \in (0,1)$ such that for all $\zeta > \zeta'$, and for every state s, a schedule maximising the expected average reward in t in $\mathcal{M}^\zeta$ is (1) an optimal schedule in the product CTMDP $\mathcal{M} \times \mathcal{A}$ from s for satisfying the ω-regular objective ϕ. Further, since $\mathcal{A}$ is a GFM, we have that (2) σ induces an optimal schedule for the CTMDP $\mathcal{M}$ from s with objective ϕ.*

From the above theorem, we have that for a large ζ value, a schedule maximising the expected average reward in $\mathcal{M}^\zeta$ also maximizes the probability of satisfying the ω-regular property in $\mathcal{M} \times \mathcal{A}$. Therefore, when the CTMDP is known, the problem of satisfaction semantics of an ω-regular property is reduced to an expected average reward objective.

The case of unknown CTMDP. Recall that we consider a CTMDP model with unknown rate and transition structure. For such unknown CTMDP models, an RL algorithm cannot construct the product $\mathcal{M} \times \mathcal{A}$ explicitly. From Theorem 3, we can conclude that any schedule maximising the expected average reward that is accrued by visiting the sink state t in $\mathcal{M}^\zeta$ where $\zeta > \zeta'$ for some $\zeta' \in (0,1)$ also maximizes the probability of satisfying the ω-regular objective ϕ in $\mathcal{M} \times \mathcal{A}$. This leads to a very simple model-free RL algorithm which does not require the augmented product CTMDP $\mathcal{M}^\zeta$ to be constructed explicitly. We define the following reward function rew' to be used by the RL algorithm:

$$rew'((s,q),a) = \begin{cases} 1 & \text{with probability } 1-\zeta \text{ if } (s,q) \text{ is} \\ & \text{accepting} \\ 0 & \text{otherwise} \end{cases}$$

Recall that in the augmented product $\mathcal{M}^\zeta$, for each action from an accepting state, we add a transition to sink state t with probability $1-\zeta$, and give a reward of 1 for staying in t per unit time. The RL algorithm simulates this in the following way: When a transition from an accepting state is visited, the learning agent tosses a biased coin and obtains a reward of 1 with probability $1-\zeta$. Therefore, any schedule maximising the expected average reward w.r.t. rew' also maximizes the probability of satisfying the objective. As Theorem 2 shows the existence of Blackwell optimal schedules in CTMDPs, we can conclude that for a high enough discount factor, any off-the-shelf model-free RL algorithm for CTMDP maximising the expected discounted reward gives an optimal schedule maximising the satisfaction of ϕ. A pseudocode of our algorithm is given in the appendix of (Falah, Guha, and Trivedi 2023).

RL for Expectation Semantics

We study the expectation semantics of ω-regular objective and show that the problem can be reduced to maximising the expected average reward problem in CTMDPs. Using Theorem 2, this reduces to maximising expected discounted reward for a large discount factor. We then describe the corresponding reward machine to maximize the *expected satisfaction time* in the good states.

Reduction to Average Reward. For an ω-regular objective ϕ, let $\mathcal{A}$ be a **GFM** corresponding to ϕ with a set F of Büchi accepting states. Let $\mathcal{M}$ be a CTMDP and $\mathcal{M} \times \mathcal{A}$ be the product CTMDP of $\mathcal{M}$ and $\mathcal{A}$. For a state s in $\mathcal{M} \times \mathcal{A}$, we define the *expected satisfaction time* of a schedule σ from starting state s as:

$$\mathsf{ESat}_\sigma^{\mathcal{M}\times\mathcal{A}}(s) = \mathbb{E}_\sigma^{\mathcal{M}\times\mathcal{A}}(s)\left\{\liminf_{n\to\infty} \frac{\sum_{i=1}^{n}[X_i \in F^\times]\cdot D_i}{T_n}\right\}.$$

It gives the long-run expected average time spent in the accepting states. The reward rate function $r' : S \to \{0,1\}$ for $\mathcal{M} \times \mathcal{A}$ is defined such that $r'(s) = 1$ if $s \in F^\times$, and $r'(s) = 0$, otherwise. Thus the reward is $r'(s) \cdot t = t$ for $s \in F^\times$ if t time is spent in s. The following lemma gives an equivalence between the expected satisfaction time and expected average reward obtained in $\mathcal{M} \times \mathcal{A}$.

Lemma 4. *For a product CTMDP $\mathcal{M} \times \mathcal{A}$ where $\mathcal{A}$ is a **GFM** for an ω-regular objective and for a schedule σ, the expected average reward obtained w.r.t. the reward function r' is equal to the expected satisfaction time in $(\mathcal{M} \times \mathcal{A})$ and there exists a pure schedule that maximizes this.*

Using the results from Lemma 4 and Theorem 2, we can conclude that a schedule maximising the discounted reward objective for a large discount factor in $\mathcal{M} \times \mathcal{A}$ with reward function r' also maximizes the expected satisfaction time.

Algorithm for Expectation Semantics. Here, we provide a brief description of the algorithm. The Q-function is defined on the states of the product CTMDP, i.e, $\mathcal{Q}_f : (\mathsf{S} \times Q) \times \mathsf{Act} \to \mathbb{R}$ where S is the set of states of the CTMDP $\mathcal{M}$ and Q is the set of states of the GFM A. Initially, the state space is unknown to the agent and the agent will have information only on the initial state. States seen are stored in a Q-table where the Q-value of the state is stored. The initial value of a state in the Q-table is zero. The number of episodes to be conducted and the length of each episode are defined by the user, let these be denoted by k and $eplen$ respectively. In each episode, the RL agent picks an action from its current state in the CTMDP according to the RL schedule and observes the next state and the time spent in the current state. It also picks the transition in the GFM based on the observed state in the CTMDP. For each transition taken, the reward obtained is based on the reward function r'. The Q-function is updated according to the Q-learning rule defined in Section . An episode ends when the length of the episode reaches $eplen$. After the completion of k episodes, we obtain a schedule σ by choosing the action that gives the highest Q-value from each state. The schedule learnt by the algorithm converges to an optimal schedule as the number of training episodes tend to infinity. A pseudocode of the algorithm is provided in (Falah, Guha, and Trivedi 2023).

Name	states	prod.	Sat. Prob.	Est. Sat.	Time 1	Exp. Val.	Est. Exp.	Time 2
RiskReward	4	8	1	1	1.713	0.9	0.9	0.967
DynamicPM-tt_3_qs_2	816	825	1	1	3.586	1	1	3.62
QS-lqs_1_rqs_1_jt_2	266	282	1	1	3.401	1	1	3.486
QS-lqs_1_rqs_1_jt_5	3977	4152	1	1	5.482	1	1	5.524
QS-lqs_2_rqs_2_jt_3	11045	24672	1	1	15.158	1	1	15.395
ftwc_001_mrmc	82	122	0.999779	0.999779	94.288	0.999779	0.999779	98.256
PollingSystem-jt1_qs4	348	352	1	1	3.423	1	1	3.421
PollingSystem-jt1_qs7	1002	1006	1	1	3.576	1	1	3.580
ErlangStages-k500_r10	508	509	1	1	112.581	1	1	312.86
SJS-procn_6_jobn_2	17	21	1	1	3.253	1	1	3.257
SJS-procn_2_jobn_6	7393	7405	1	1	4.336	1	1	4.234

Table 1: Q-learning results. The default values of the learner hyperparameters are: $\zeta = 0.99$ (for satisfaction semantics), $\epsilon = 0.1$ (used in picking ϵ-greedy actions in Q-learning), $\beta = 0.01$ (learning rate), tol$= 0.01$ (tolerance for numerical approximation), ep-l$= 300$ (episode length), and ep-n$= 20000$ (episode numbers). We used ep-l = 3000 and ep-n = 200000 for ErlangStages-k500_r10 to get convergence. Times are in seconds.

Experimental Evaluation

We implemented the reward schemes described in the previous sections in a C++-based tool MUNGOJERRIE (Hahn et al. 2021) which reads CTMDPs described in the PRISM language (Kwiatkowska, Norman, and Parker 2011) and ω-regular automata written in the *Hanoi Omega Automata* format (Babiak et al. 2015). Our implementation provides an Openai-gym (Brockman et al. 2016) style interface for RL algorithms and supports probabilistic model checking for CTMDPs based on uniformization.

Table 1 shows the evaluation of our algorithms on a set of CTMDP benchmarks from the Quantitative Verification Benchmark set (https://qcomp.org). RiskReward is based on the model in Section with $\lambda(0, b) = 10$ and $r = 9$. DynamicPM-tt_3_qs_2 models dynamic power management problem based on (Simunic et al. 2000). Queuing System (QS) models QS-lqs_i_rqs_j_jt_k are based on a CTMDP modelling of queuing systems with arrival rate i, service rate j, and jump rate k as the key parameters. ftwc_001_mrmc models consist of two networks of n workstations where each network is interconnected by a switch communicating via a backbone. The components may fail arbitrarily, but can only be repaired one at a time. The initial state is where all components are functioning, and the goal state is where in both networks either all the workstations or all the switches are broken. The Polling System examples PollingSystem-jt1_qsj consist of j stations and 1 server. Here, the incoming requests of j types are buffered in queues of size k each, until they are processed by the server and delivered to their station. The system starts in a state with all the queues being nearly full. We consider 2 goal conditions: (i) all the queues are empty and (ii) one of the queues is empty. The ErlangStages-k500_r10 model has two different paths to reach the goal state: a fast but risky path or a slow but sure path. The slow path is an Erlang chain of length 500 and rate 10. The objective is to check which path gives faster reachability to a safe target state. The stochastic job scheduling benchmarks SJS-procn_i_jobn_j model multiple processors (i) and independent jobs (j) with a goal of job completion.

The results are summarized in Table 1. For each model, we provide the number of states in the CTMDP (*states*) and in the product CTMDP (*prod*), the probability of satisfaction (*Sat. Prob.*) for the satisfaction semantics, estimated probability by the RL algorithm (*Est. Sat.*), and time (*Time 1*) spent in learning that schedule. The value (expected satisfaction time) for the expectation semantics (*Exp. Val.*), estimated value obtained by the RL algorithm (*Est. Exp.*), and the learning time (*Time 2*) for the expectation semantics are provided next. All of our timings and values are averaged over three runs with randomly chosen seeds. We use the hyperparameters as shown in the caption of Table 1.

Our experimental results demonstrate that the proposed RL algorithms are effective in handling medium sized CTMDPs. For both semantics, one may note that the RL algorithms efficiently estimate the optimal values and compute the optimal schedules. The results for both the semantics for the benchmarks from the QComp set are similar as they were used for reachability objective with the target states being terminal states havng self loops.

Conclusion

Continuous-time MDPs are canonical models to express nondeterministic and stochastic behavior under dense-time semantics. Reinforcement learning (RL) provides a sampling-based method to compute an optimal schedule in the absence of an explicit environment model. The RL approach for CTMDPs has recently received considerable attention (Guo and Zhang 2016; Rabe and Schewe 2013). Our work enabled the specification of learning objectives in CTMDPs as ω-regular specifications. To accommodate temporal modelling, we consider two semantics of ω-regular specifications (that include LTL objectives) and provide translations to scalar reward forms amenable for model-free reinforcement learning. We believe that this work will open doors to study and develop model-free reinforcement learning for continuous-time models that go beyond CTMDPs and allow temporal constraints on planner's choices and residence-time requirements.

Acknowledgements

This work is partially supported by DST-SERB grant SRG/2021/000466 and by the National Science Foundation (NSF) grant CCF-2009022 and by NSF CAREER award CCF-2146563.

References

Babiak, T.; Blahoudek, F.; Duret-Lutz, A.; Klein, J.; Křetínský, J.; Müller, D.; Parker, D.; and Strejček, J. 2015. The Hanoi Omega-Automata Format. In *Computer Aided Verification*, 479–486. LNCS 9206.

Baier, C.; and Katoen, J. 2008. *Principles of model checking*. MIT Press.

Baykal-Gürsoy, M. 2011. *Semi-Markov Decision Processes*. Hoboken, NJ, USA: John Wiley & Sons, Inc.

Bradtke, S. J.; and Duff, M. O. 1994. Reinforcement Learning Methods for Continuous-Time Markov Decision Problems. In *NIPS Conference*, 393–400. MIT Press.

Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. OpenAI Gym. *CoRR*, abs/1606.01540.

Camacho, A.; Icarte, R. T.; Klassen, T. Q.; Valenzano, R. A.; and McIlraith, S. A. 2019. LTL and Beyond: Formal Languages for Reward Function Specification in Reinforcement Learning. In *IJCAI*, volume 19, 6065–6073.

Das, T. K.; Gosavi, A.; Mahadevan, S.; and Marchalleck, N. 1999. Solving semi-Markov decision problems using average reward reinforcement learning. *Management Science*, 45(4): 560–574.

De Alfaro, L. 1998. Formal Verification of Probabilistic Systems. Technical report, Stanford, CA, USA.

Falah, A.; Guha, S.; and Trivedi, A. 2023. Reinforcement Learning for Omega-Regular Specifications on Continuous-Time MDP. *CoRR*, abs/2303.09528.

Feinberg, E.; and Shwartz, A., eds. 2002. *Handbook of Markov Decision Processes Methods and Applications*. Kluwer International Series.

Goodfellow, I.; Bengio, Y.; and Courville, A. 2016. *Deep Learning*. MIT Press.

Guo, X.; and Hernández-Lerma, O. 2009. Continuous-time Markov decision processes. In *Continuous-Time Markov Decision Processes*, 9–18. Springer.

Guo, X.; and Zhang, Y. 2016. Optimality of Mixed Policies for Average Continuous-Time Markov Decision Processes with Constraints. *Math. Oper. Res.*, 41(4): 1276–1296.

Hahn, E. M.; Perez, M.; Schewe, S.; Somenzi, F.; Trivedi, A.; and Wojtczak, D. 2019. Omega-Regular Objectives in Model-Free Reinforcement Learning. In *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 11427 of *LNCS*, 395–412. Springer.

Hahn, E. M.; Perez, M.; Schewe, S.; Somenzi, F.; Trivedi, A.; and Wojtczak, D. 2020. Good-for-MDPs Automata for Probabilistic Analysis and Reinforcement Learning. In *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, 306–323.

Hahn, E. M.; Perez, M.; Schewe, S.; Somenzi, F.; Trivedi, A.; and Wojtczak, D. 2021. Mungojerrie: Reinforcement Learning of Linear-Time Objectives. To appear in TACAS'23, arXiv:2106.09161.

Icarte, R. T.; Klassen, T.; Valenzano, R.; and McIlraith, S. 2018. Using reward machines for high-level task specification and decomposition in reinforcement learning. In *ICML*, 2107–2116. PMLR.

Icarte, R. T.; Klassen, T. Q.; Valenzano, R.; and McIlraith, S. A. 2022. Reward machines: Exploiting reward function structure in reinforcement learning. *Journal of Artificial Intelligence Research*, 73: 173–208.

Kwiatkowska, M.; Norman, G.; and Parker, D. 2011. PRISM 4.0: Verification of Probabilistic Real-time Systems. In *Computer Aided Verification*, 585–591. LNCS 6806.

Li, X.; Vasile, C.-I.; and Belta, C. 2017. Reinforcement learning with temporal logic rewards. In *International Conference on Intelligent Robots*, 3834–3839. IEEE.

Mnih, V.; et al. 2015. Human-level control through reinforcement learning. *Nature*, 518: 529–533.

Oura, R.; and Ushio, T. 2022. Learning-based Bounded Synthesis for Semi-MDPs with LTL Specifications. *IEEE Control Systems Letters*, 6: 2557–2562.

Puterman, M. L. 2014. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Rabe, M. N.; and Schewe, S. 2013. Optimal time-abstract schedulers for CTMDPs and continuous-time Markov games. *Theor. Comput. Sci.*, 467: 53–67.

Sadigh, D.; Kim, E. S.; Coogan, S.; Sastry, S. S.; and Seshia, S. A. 2014. A learning based approach to control synthesis of Markov decision processes for linear temporal logic specifications. In *IEEE Conference on Decision and Control (CDC)*, 1091–1096.

Sickert, S.; Esparza, J.; Jaax, S.; and Křetínskỳ, J. 2016. Limit-deterministic Büchi automata for linear temporal logic. In *Computer Aided Verification*, 312–332. Springer.

Silver, D.; et al. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529: 484–489.

Simunic, T.; Benini, L.; Glynn, P.; and De Micheli, G. 2000. Dynamic Power Management for Portable Systems. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, 11–19. ISBN 1581131976.

Strehl, A. L.; Li, L.; Wiewiora, E.; Langford, J.; and Littman, M. L. 2006. PAC model-free reinforcement learning. In *ICML*, 881–888.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement Learnging: An Introduction*. MIT Press, second edition.

Thomas, W. 2009. Facets of synthesis: Revisiting church's problem. In *International Conference on Foundations of Software Science and Computational Structures*, 1–14. Springer.

Vardi, M. Y.; and Wolper, P. 1986. Automata-Theoretic Techniques for Modal Logics of Programs. *J. Comput. Syst. Sci.*, 32(2): 183–221.

Watkins, C. J. C. H.; and Dayan, P. 1992. Technical Note Q-Learning. *Machine Learning*, 8: 279–292.