Formal Controller Synthesis for Continuous-Space MDPs via Model-Free Reinforcement Learning

1st Abolfazl Lavaei
Department of Computer Science
Ludwig Maximilian University of Munich
Munich, Germany
lavaei@lmu.de

2rd Fabio Somenzi
Electrical, Computer & Energy Engineering
University of Colorado Boulder
Boulder, USA
fabio@colorado.edu

3th Sadegh Soudjani
School of Computing
Newcastle University
Newcastle, United Kingdom
sadegh.soudjani@newcastle.ac.uk

4th Ashutosh Trivedi
Department of Computer Science
University of Colorado Boulder
Boulder, USA
ashutosh.trivedi@colorado.edu

5th Majid Zamani
Department of Computer Science
University of Colorado Boulder, Boulder, USA
Ludwig Maximilian University of Munich, Munich, Germany
majid.zamani@colorado.edu

***Abstract*—A novel reinforcement learning scheme to synthesize policies for continuous-space Markov decision processes (MDPs) is proposed. This scheme enables one to apply model-free, off-the-shelf reinforcement learning algorithms for finite MDPs to compute optimal strategies for the corresponding continuous-space MDPs without explicitly constructing the finite-state abstraction. The proposed approach is based on abstracting the system with a finite MDP (without constructing it explicitly) with *unknown* transition probabilities, synthesizing strategies over the abstract MDP, and then mapping the results back over the concrete continuous-space MDP with *approximate optimality guarantees*. The properties of interest for the system belong to a fragment of linear temporal logic, known as syntactically co-safe linear temporal logic (scLTL), and the synthesis requirement is to maximize the probability of satisfaction within a given bounded time horizon. A key contribution of the paper is to leverage the classical convergence results for reinforcement learning on finite MDPs and provide control strategies maximizing the probability of satisfaction over unknown, continuous-space MDPs while providing probabilistic closeness guarantees. Automata-based reward functions are often sparse; we present a novel potential-based reward shaping technique to produce dense rewards to speed up learning. The effectiveness of the proposed approach is demonstrated by applying it to three physical benchmarks concerning the regulation of a room's temperature, control of a road traffic cell, and of a 7-dimensional nonlinear model of a BMW 320i car.**

***Index Terms*—Model-Free Reinforcement Learning, Continuous-Space MDPs, Formal Controller Synthesis.**

I. Introduction

Motivations. Control systems with stochastic uncertainty can be modeled as Markov decision processes (MDPs) over uncountable state and action spaces. These stochastic models have received significant attentions as an important modeling framework describing many engineering systems; they play significant roles in many safety-critical applications including power grids and traffic networks. Automated controller synthesis [1] for general MDPs to achieve some high-level specifications, e.g., those expressed as linear temporal logic (LTL) formulae [2], is inherently challenging due to its computational complexity and uncountable sets of states and actions. Closed-form computation of optimal policies for MDPs over uncountable spaces is not available in general. One promising approach is to first approximate these models by simpler ones with finite state sets, perform analysis and synthesis over the abstract models (using algorithms from formal methods [1]), and translate the results back over the original system, while providing guaranteed error bounds in the detour process.

Related Literature. There have been several results, proposed in the past few years, on abstraction-based synthesis of continuous-space MDPs. Existing results include construction of finite MDPs for formal verification and synthesis [3] and the extension of such techniques to infinite horizon properties [4] under some strong assumptions over the dynamics. Algorithmic construction of the abstract models and performing formal synthesis over them are studied in [5], [6]. Safety verification and formal synthesis of stochastic systems are respectively studied in [7] and [8] using so-called control barrier certificates. Although the proposed approaches in [7] and [8] do not need the state set discretization, they require knowing precisely the probabilistic evolution of states in models which may not be known in general.

To construct finite MDPs of continuous-space ones with guaranteed error bounds between them, we need to establish some sort of similarity relations between them. Similarity relations over finite-state stochastic systems have been studied, either via exact notions of probabilistic (bi)simulation relations [9], [10] or approximate versions [11]. Similarity relations for models with general, uncountable state spaces have also been proposed in the literature. These relations either depend on stability requirements on model outputs via martingale theory or the contractivity analysis [12] or enforce structural abstractions of models by exploiting continuity

This work was supported in part by the H2020 ERC Starting Grant AutoCPS (grant agreement No. 804639).

conditions on their probability laws [13]. A new bisimilarity relation is proposed in [14], [15] based on the joint probability distribution of two models and enables combining model reduction together with space discretization.

Unfortunately, construction of finite MDPs, studied in the aforementioned literatures, suffers severely from the so-called *curse of dimensionality*: the computational complexity of constructing finite MDPs grows exponentially as the number of state variables increases. In addition, one needs to know precise models of continuous-space MDPs to construct those finite abstractions and, hence, the proposed approaches in the relevant literature are not applicable in the settings where the transition structure is unknown. These challenges motivated us to employ reinforcement learning for the controller synthesis of such complex systems.

Reinforcement learning (RL) [16] is an approach to sequential decision making in which agents rely on reward signals to choose actions aimed at achieving prescribed objectives. Model-free reinforcement learning [17] refers to techniques that are asymptotically space-efficient because they do not store the probabilistic transition structure of the environment. These techniques include algorithms like TD(λ) [18] and Q-learning [19] as well as their extensions to deep neural networks such as deep deterministic policy gradient (DDPG) [20] and neural-fitted Q-iterations [21]. Model-free reinforcement learning has achieved performance comparable to that of human experts in video and board games [22]–[24]. This success has motivated extensions of reinforcement learning to the control of safety-critical systems [20], [25] in spite of a lack of theoretical convergence guarantees of reinforcement learning for general continuous state spaces [26].

Main contribution. By utilizing a closeness guarantee between probabilities of satisfaction by the unknown continuous-space MDP and by its finite abstraction which can be controlled a-priori, and leveraging the classical convergence results for reinforcement learning on finite-state MDPs, we provide, for the first time, a reinforcement learning approach for MDPs with uncountable state sets while providing convergence guarantees. In particular, this approach enables us to apply model-free, off-the-shelf reinforcement learning algorithms to compute ε-optimal strategies for continuous-space MDPs with a precision ε that is defined a-priori and without explicitly constructing finite abstractions. Another key contribution of the paper is a novel potential-based reward shaping [27] technique to produce dense rewards that is based on the structure of the property automaton. Although the techniques presented in this paper can be adapted to model-based RL, the experiments presented in this work deal with model-free RL.

Recent Works. A model-free reinforcement learning framework for synthesizing policies for unknown, and possibly continuous-state, MDPs is recently presented in [28]–[30]. The proposed approaches in [28]–[30] provide theoretical guarantees only when the MDP has the finite number of states, and the corresponding results for continuous-state MDPs are empirically illustrated. The results in [31] provide a reinforcement learning approach for *deterministic* continuous-space control systems where the closeness between finite approximations and concrete models are only guaranteed asymptotically, rather than according to some formal relations that are in the end required to ensure the correspondence of controllers for temporal logic specifications over model trajectories. The results in [32] provide deterministic policy gradient algorithms for MDPs with continuous state and action spaces using reinforcement learning, but without providing any quantitative guarantee on the optimality of synthesized policies for original MDPs. In contrast, we utilize here a closeness guarantee between probabilities of satisfaction by the unknown continuous-space MDP and by its finite abstraction to compute ε-optimal strategies for original systems using the reinforcement learning with a-priori defined precision ε.

Organization. The rest of this paper is organized as follows. In the next section we introduce background definitions and notations. Then we formulate the main problem in the section afterward. In that section, in particular, we propose closeness guarantees between probabilities of satisfaction by continuous-space MDPs and their finite-state counterparts and its connection to the classical convergence results for reinforcement learning on finite-state MDPs. Finally, to demonstrate the effectiveness of the proposed results, we apply our approaches to several physical benchmarks in the last section.

II. Preliminaries

As usual, we write $\mathbb{N}$ and $\mathbb{N}_{>0}$ for sets of nonnegative and positive integers. Similarly, we write $\mathbb{R}$, $\mathbb{R}_{>0}$, and $\mathbb{R}_{\geq 0}$ for sets of reals, positive and nonnegative reals, respectively. For a set of N vectors, $x_1 \in \mathbb{R}^{n_1}, \ldots, x_N \in \mathbb{R}^{n_N}$, we write $[x_1; \ldots; x_N]$ to denote the corresponding vector of dimension $\sum_i n_i$. Given a vector $x \in \mathbb{R}^n$ we write $\|x\|$ for its Euclidean norm and for $a \in \mathbb{R}$ we write $|a|$ for its absolute value.

A *discrete probability distribution*, or just distribution, over a (possibly uncountable) set X is a function $d : X \to [0, 1]$ such that $\sum_{x \in X} d(x) = 1$ and $supp(d) = \{x \in X \mid d(x) > 0\}$ is at most countable. Let $\mathcal{D}(X)$ denote the set of all discrete distributions over X. We consider a probability space $(\Omega, \mathcal{F}_\Omega, \mathbb{P}_\Omega)$, where Ω is the sample space, $\mathcal{F}_\Omega$ is a sigma-algebra on Ω comprising subsets of Ω as events, and $\mathbb{P}_\Omega$ is a probability measure that assigns probabilities to events. We assume that random variables introduced in this article are measurable functions of the form $X : (\Omega, \mathcal{F}_\Omega) \to (S_X, \mathcal{F}_X)$. Any random variable X induces a probability measure on its space $(S_X, \mathcal{F}_X)$ as $Prob\{A\} = \mathbb{P}_\Omega\{X^{-1}(A)\}$ for any $A \in \mathcal{F}_X$. When clear from the context, we use the probability measure on $(S_X, \mathcal{F}_X)$ without mentioning the probability space and the function X.

A topological space S is called a *Borel space* if it is homeomorphic to a Borel subset of a Polish space (i.e., a separable and completely metrizable space). Examples of a Borel space are Euclidean space $\mathbb{R}^n$, its Borel subsets endowed with a subspace topology, as well as hybrid spaces. Any Borel space S is assumed to be endowed with a Borel sigma-algebra, which is denoted by $\mathcal{B}(S)$. We say that a map $f : S \to Y$ is measurable whenever it is Borel measurable.

A. Discrete-Time Stochastic Control Systems

Definition 1. *A discrete-time stochastic control system (dt-SCS) is a tuple*

$$\Sigma = (X, U, \varsigma, f), \tag{1}$$

where

- *$X \subseteq \mathbb{R}^n$ is a Borel space as the state space of the system. We denote by $(X, \mathcal{B}(X))$ the measurable space with $\mathcal{B}(X)$ being the Borel sigma-algebra X;*
- *U is the input space of the system;*
- *ς is a sequence of independent and identically distributed (i.i.d.) random variables from a sample space Ω to the set V_ς, namely $\varsigma := \{\varsigma(k) : \Omega \to V_\varsigma,\ k \in \mathbb{N}\}$;*
- *$f : X \times U \times V_\varsigma \to X$ is a measurable function characterizing the state evolution of Σ.*

The evolution of the state of dt-SCS Σ for an initial state $x(0) \in X$ and an input sequence $\{\nu(k) : \Omega \to U,\ k \in \mathbb{N}\}$ is described as:

$$x(k+1) = f(x(k), \nu(k), \varsigma(k)). \tag{2}$$

Remark 2. *The input space U of a dt-SCS Σ is in general a continuous Borel space, e.g., a subset of $\mathbb{R}^m$. Since any input sequence will be implemented by a digital controller, w.l.o.g. we assume that the input space U is finite.*

We define *Markov policies* to control the system in (1).

Definition 3. *For the dt-SCS Σ in* (1)*, a Markov policy is a sequence $\rho = (\rho_0, \rho_1, \rho_2, \ldots)$ of universally measurable stochastic kernels ρ_n [33], each defined on the input space U given X such that for all $x_n \in X$, $\rho_n(U \,|\, x_n) = 1$. The class of all such Markov policies is denoted by $\bar{\Pi}_M$.*

We associate to U the set $\mathcal{U}$ to be the collection of sequences $\{\nu(k) : \Omega \to U,\ k \in \mathbb{N}\}$, in which $\nu(k)$ is independent of $\varsigma(t)$ for any $k, t \in \mathbb{N}$ and $t \geq k$. The random sequence $x_{a\nu} : \Omega \times \mathbb{N} \to X$ satisfying (2) for any initial state $a \in X$, and $\nu(\cdot) \in \mathcal{U}$ is called the *solution process* of Σ under the input ν and the initial state a.

B. Requirement Specification in scLTL

Formal requirements provide the rigorous and unambiguous formalism to express requirements over MDPs [1]. A common way to describe formal requirements is by using automata-based specifications or logic-based specifications using formulae in, for instance, linear temporal logic (LTL). For example, consider a dt-SCS Σ in (1) and a measurable target set $\mathsf{B} \subset X$. We say that a state trajectory $\{x(k)\}_{k \geq 0}$ reaches a target set B within time interval $[0, T] \subset \mathbb{N}$, if there exists a $k \in [0, T]$ such that $x(k) \in \mathsf{B}$. This bounded reaching of B is denoted by $\lozenge^{\leq T}\{x \in \mathsf{B}\}$ or briefly $\lozenge^{\leq T}\mathsf{B}$. For $T \to \infty$, we denote the reachability property as $\lozenge\mathsf{B}$, i.e., eventually B. For a dt-SCS Σ with a policy ρ, we want to compute the probability that a state trajectory reaches B within the time horizon $T \in \mathbb{N}$, i.e., $\mathbb{P}(\lozenge^{\leq T}\mathsf{B})$. The *reachability probability* is the probability that the target set B is eventually reached and is denoted by $\mathbb{P}(\lozenge\mathsf{B})$. In this paper, we deal with properties more complex than simple reachability property.

Finite Automata. A *deterministic finite automaton* (DFA) is a tuple $\mathcal{A} = (Q, \Sigma_\mathsf{a}, \mathsf{t}, q_0, F_\mathsf{a})$ where Q is a finite set of states, Σ_a is an alphabet, $\mathsf{t} : Q \times \Sigma_\mathsf{a} \to Q$ is a transition function, $q_0 \in Q$ is the initial state, and $F_\mathsf{a} \subseteq Q$ are accepting states. We write λ for the empty string and Σ_a^* for the set of all strings over Σ_a. The extended transition function $\hat{\mathsf{t}} : Q \times \Sigma_\mathsf{a}^* \to Q$ (transition function extended to summarize the effect of reading a string) can be defined as:

$$\hat{\mathsf{t}}(q, \bar{w}) = \begin{cases} q, & \text{if } \bar{w} = \lambda, \\ \mathsf{t}(\hat{\mathsf{t}}(q, x), a), & \text{if } \bar{w} = xa \text{ for } x \in \Sigma_\mathsf{a}^* \text{ and } a \in \Sigma_\mathsf{a}. \end{cases}$$

The language $\mathcal{L}(\mathcal{A})$ accepted by a DFA $\mathcal{A}$ is defined as $\mathcal{L}(\mathcal{A}) = \{\bar{w} : \hat{\mathsf{t}}(q_0, \bar{w}) \in F_\mathsf{a}\}$. DFAs are well-established models to express regular specifications over finite words. DFAs can also be interpreted over ω-words: an ω-word is accepted if there is a prefix that is accepted by the DFA. Among others, DFAs are expressive enough to capture syntactically a co-safe fragment of linear temporal logic (LTL) defined next.

Linear Temporal Logic. Consider a set of atomic propositions AP and the alphabet $\Sigma_\mathsf{a} := 2^{AP}$. Let $\omega = \omega(0), \omega(1), \omega(2), \ldots \in \Sigma_\mathsf{a}^\mathbb{N}$ be an infinite word, that is, a string composed of letters from Σ_a. We are interested in those atomic propositions that are relevant to the dt-SCS via a measurable labeling function L from the state space to the alphabet as $\mathsf{L} : X \to \Sigma_\mathsf{a}$. State trajectories $\{x(k)\}_{k \geq 0} \in X^\mathbb{N}$ can be readily mapped to the set of infinite words $\Sigma_\mathsf{a}^\mathbb{N}$, as $\omega = \mathsf{L}(\{x(k)\}_{k \geq 0}) := \{\omega \in \Sigma_\mathsf{a}^\mathbb{N} \,|\, \omega(k) = \mathsf{L}(x(k))\}$. Consider LTL properties with the syntax [1]

$$\phi ::= \text{true} \mid p \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \bigcirc\phi \mid \phi_1 \,\mathsf{U}\, \phi_2.$$

Let $\omega_k = \omega(k), \omega(k+1), \omega(k+2), \ldots$ be a subsequence (suffix) of ω, then the satisfaction relation between ω and a property ϕ, expressed in LTL, is denoted by $\omega \vDash \phi$ (or equivalently $\omega_0 \vDash \phi$). Semantics of the satisfaction relation are defined recursively over ω_k and the syntax of the LTL formula ϕ. An atomic proposition $p \in AP$ is satisfied by ω_k, i.e., $\omega_k \vDash p$, iff $p \in \omega(k)$. Furthermore, $\omega_k \vDash \neg\phi$ if $\omega_k \nvDash \phi$ and we say that $\omega_k \vDash \phi_1 \wedge \phi_2$ if $\omega_k \vDash \phi_1$ and $\omega_k \vDash \phi_2$. The next operator $\omega_k \vDash \bigcirc\phi$ holds if the property holds at the next time instance $\omega_{k+1} \vDash \phi$. We denote by $\bigcirc^j$, $j \in \mathbb{N}$, j times composition of the next operator. With a slight abuse of the notation, one has $\bigcirc^0\phi = \phi$ for any property ϕ. The temporal until operator $\omega_k \vDash \phi_1 \,\mathsf{U}\, \phi_2$ holds if $\exists i \in \mathbb{N} : \omega_{k+i} \vDash \phi_2$, and $\forall j \in \mathbb{N} : 0 \leq j < i, \omega_{k+j} \vDash \phi_1$. Based on these semantics, disjunction ($\vee$) can be defined by $\omega_k \vDash \phi_1 \vee \phi_2 \Leftrightarrow \omega_k \vDash \neg(\neg\phi_1 \wedge \neg\phi_2)$. This paper focuses on a fragment of LTL properties known as syntactically co-safe linear temporal logic (scLTL) [34] defined below.

Definition 4 (Syntactically Co-Safe LTL (scLTL))**.** *An scLTL over a set of atomic propositions AP is a fragment of LTL* *such that the negation operator ($\neg$) only occurs before atomic propositions characterized by the following grammar:*

$$\phi ::= p \,|\, \neg p \,|\, \phi_1 \vee \phi_2 \,|\, \phi_1 \wedge \phi_2 \,|\, \bigcirc\phi \,|\, \phi_1 \,\mathsf{U}\, \phi_2.$$

Even though scLTL formulas are defined over infinite words (as in LTL formulae), their satisfaction is guaranteed in the finite time [34]. Any infinite word $\omega \in \Sigma_{\mathsf{a}}^{\mathbb{N}}$ satisfying an scLTL formula ϕ has a finite word $\omega_f \in \Sigma_{\mathsf{a}}^{n}$, $n \in \mathbb{N}$, as its prefix such that all infinite words with a prefix ω_f also satisfy the formula ϕ. We denote the set of all such finite prefixes associated with an scLTL formula ϕ by $\mathcal{L}_f(\phi)$.

For verification and synthesis purposes, the scLTL properties can be compiled into a DFA $\mathcal{A}_\phi$ over the alphabet 2^{AP} such that $\mathcal{L}_f(\phi) = \mathcal{L}(\mathcal{A}_\phi)$ [34]. This construction is routine; we refer the interested reader to [34] for details of the construction of the DFA $\mathcal{A}_\phi$ from ϕ such that $\mathcal{L}(\mathcal{A}_\phi) = \mathcal{L}_f(\phi)$. The resulting DFA has the property that there is a unique accepting state and all out-going transitions from that state are self-loops. Such a DFA is also known as a *co-safety automaton*. In the rest of the paper we assume that the DFA $\mathcal{A}_\phi$ for an scLTL property ϕ is a co-safety automaton.

Given a policy ρ, the probability that a state trajectory of Σ satisfies an scLTL property ϕ over the time horizon $[0, T]$, is denoted by $\mathbb{P}(\omega_f \in \mathcal{L}(\mathcal{A}_\phi)$ s.t. $|\omega_f| \leq T+1)$, where $|\omega_f|$ is the length of ω_f [35]. The co-safety automaton for the bounded time-horizon satisfaction can be computed by unrolling the DFA for ϕ. In the rest of the paper we assume such a representation for the finite horizon satisfaction.

Remark 5. *We emphasize that there is no* closed-form solution *for computing optimal policies enforcing scLTL specifications over* continuous-space *MDPs. One can employ the approximation approaches, discussed later, to synthesize those policies which, however, suffer severely from the curse of dimensionality and require knowing precisely the probabilistic evolution of states in models. Instead, we propose in this paper, for the first time, an RL approach providing policies for unknown, continuous-space MDPs while providing* quantitative guarantees *on the satisfaction of properties.*

III. Controller Synthesis for Unknown Continuous-Space MDPs

We are interested in automatically synthesizing controllers for unknown continuous-space MDPs whose requirements are provided as scLTL specifications. Given a discrete-time stochastic control system $\Sigma = (X, U, \varsigma, f)$, where f and the distribution of ς are unknown, and given an scLTL formula ϕ, we wish to synthesize a Markov policy enforcing the property ϕ over Σ with the probability of satisfaction within a guaranteed threshold from the unknown optimal probability.

In order to provide any formal guarantee, we need to make further assumptions on the dt-SCS. In particular, we assume that the dynamical system in (2) is Lipschitz-continuous with a constant $\mathscr{H}$. Consider the dynamical system in (2) where $\varsigma(\cdot)$ is i.i.d. with a known distribution $t_\varsigma(\cdot)$. Suppose that the vector field f is continuously differentiable and the matrix $\frac{\partial f}{\partial \varsigma}$ is invertible. Then, the *implicit function theorem* guarantees the existence and uniqueness of a function $g : X \times X \times U \to V_\varsigma$ such that $\varsigma(k) = g(x(k+1), x(k), \nu(k))$. In this case, the conditional density function is:

$$t_x(x'|x, \nu) = \left|\det\left[\frac{\partial g}{\partial x'}(x', x, \nu)\right]\right| t_\varsigma(g(x', x, \nu)).$$

The Lipschitz constant $\mathscr{H}$ is specified by the dependency of the function $g(x', x, \nu)$ on the variable x. As a special case consider a nonlinear system with an additive noise

$$f(x, \nu, \varsigma) = f_a(x, \nu) + \varsigma.$$

Then the invertibility of $\frac{\partial f}{\partial \varsigma}$ is guaranteed and $g(x', x, \nu) = x' - f_a(x, \nu)$. In this case, $\mathscr{H}$ is the product of the Lipschitz constant of $t_\varsigma(\cdot)$ and $f_a(\cdot)$.

The next example provides a systematic way of computing $\mathscr{H}$ for the class of linear continuous-space MDPs with an additive noise.

Example 6. *Consider a dt-SCS Σ with linear dynamics $x(k+1) = Ax(k) + B\nu(k) + \varsigma(k)$, where $A = [a_{ij}]$ and $\varsigma(k)$ are i.i.d. for $k = 0, 1, 2, \ldots$ with a normal distribution having zero mean and covariance matrix $diag^1(\sigma_1, \ldots, \sigma_n)$. Then, one obtains $\mathscr{H} = \sum_{i,j} \dfrac{2|a_{ij}|}{\sigma_i\sqrt{2\pi}}$. Note that for the computation of the approximation error (cf. (7)), it is sufficient to know an upper bound on entries of the matrix A and a lower bound on the standard deviation of the noise.*

An alternative way of computing the Lipschitz constant $\mathscr{H}$ is to estimate it from sample trajectories of Σ. This can be done by first constructing a non-parametric estimation of the conditional density function using techniques proposed in [36] and then compute the Lipschitz constant numerically using the derivative of the estimated conditional density function.

More specifically, we can use a conditional kernel density estimation (CKDE) that puts a kernel around each data point. The main purpose of using kernels is to interpolate between the observed data in order to predict the density at the unobserved data points. CKDE provides the following estimation for the conditional density function:

$$t_x^{est}(x'|x) = \frac{\sum_{i=1}^{N_\mathsf{s}} K_{\bar{h}_1}(x' - x_i')K_{\bar{h}_2}(\|x - x_i\|)}{\sum_{i=1}^{N_\mathsf{s}} K_{\bar{h}_2}(\|x - x_i\|)}, \qquad (3)$$

where data pairs (x_i, x_i') are extracted from sample trajectories with x_i' being the observed next state for the current state x_i, $K_{\bar{h}}(y) := \frac{1}{\bar{h}^n}K(\frac{y}{\bar{h}})$, K is a kernel function, i.e., a symmetric probability distribution with a bounded variance (e.g. the Gaussian), n is the dimension of x, and $\bar{h}$ is the bandwidth controlling the kernel widths. This form is known as the Nadaraya-Watson conditional density estimator, which is consistent when $\bar{h}_1 \to 0$, $\bar{h}_2 \to 0$, and $N_\mathsf{s}\bar{h}_1\bar{h}_2 \to 0$ as $N_\mathsf{s} \to 0$ [37]. In our case, we can use (3) while making both sides also dependent on the input ν, and then compute its Lipschitz constant numerically. The numerical computation

[1]$diag(\sigma_1, \ldots, \sigma_n)$ is a diagonal matrix with $\sigma_1, \ldots, \sigma_n$ as its entries.

involves taking the derivative w.r.t. x, then its norm, and finally maximizing over both x, x'. Note that we do not need the best possible Lipschitz constant: any upper bound is also sufficient but at the cost of making the formulated errors more conservative (cf. errors (7)-(8) in Theorem 8).

Now we have all required ingredients to state the main problem we address in this paper.

Problem 7. *Let ϕ be an scLTL formula and $\Sigma = (X, U, \varsigma, f)$ a continuous-space MDP, where f and the distribution of ς are unknown, but the Lipschitz constant $\mathscr{H}$ is known. Synthesize a Markov policy that satisfies the property ϕ over Σ with probability within a-priori defined threshold ε from the unknown optimal probability.*

To present our solution to this problem, we first present a technical result connecting continuous-space MDPs with corresponding finite MDP abstractions. We then exploit this result to provide a reinforcement learning-based solution to Problem 7. We emphasize again that we do not construct explicitly finite abstractions of continuous-space MDPs in this work. In fact, we cannot construct them because the dynamics of continuous-space MDPs are unknown.

A. Abstraction of dt-SCS Σ by a Finite MDP

A dt-SCS Σ in (1) can be *equivalently* represented as a Markov decision process (MDP) [38, Proposition 7.6]

$$\Sigma = (X, U, T_{\mathsf{x}}),$$

where the map $T_{\mathsf{x}} : \mathcal{B}(X) \times X \times U \rightarrow [0, 1]$, is a conditional stochastic kernel that assigns to any $x \in X$, and $\nu \in U$, a probability measure $T_{\mathsf{x}}(\cdot|x, \nu)$ on the measurable space $(X, \mathcal{B}(X))$ so that for any set $\mathcal{A} \in \mathcal{B}(X)$,

$$\mathbb{P}(x(k+1) \in \mathcal{A} \,|\, x(k), \nu(k)) = \int_{\mathcal{A}} T_{\mathsf{x}}(dx' \,|\, x(k), \nu(k)).$$

For given input $\nu(\cdot)$, the stochastic kernel T_{x} captures the evolution of the state of Σ and can be uniquely determined by the pair (ς, f) from (1). In other words, T_{x} contains the information of the function f and the distribution of the noise $\varsigma(\cdot)$ in the dynamical representation.

Now we approximate a dt-SCS Σ with a *finite* $\widehat{\Sigma}$ using an abstraction algorithm. The algorithm first constructs a finite partition of the state space $X = \cup_i \mathsf{X}_i$. Then representative points $\bar{x}_i \in \mathsf{X}_i$ are selected as abstract states. Given a dt-SCS $\Sigma = (X, U, \varsigma, f)$, the constructed finite MDP $\widehat{\Sigma}$ is

$$\widehat{\Sigma} = (\hat{X}, \hat{U}, \varsigma, \hat{f}), \tag{4}$$

where $\hat{X} = \{\bar{x}_i, i = 1, \ldots, n_x\}$, a finite subset of X, and $\hat{U} := U$ are finite state and input sets of the MDP $\widehat{\Sigma}$. Moreover, $\hat{f} : \hat{X} \times \hat{U} \times V_\varsigma \rightarrow \hat{X}$ is defined as $\hat{f}(\hat{x}, \hat{\nu}, \varsigma) = \Pi_x(f(\hat{x}, \hat{\nu}, \varsigma))$, where $\Pi_x : X \rightarrow \hat{X}$ is the map that assigns to any $x \in X$, the representative point $\bar{x} \in \hat{X}$ of the corresponding partition set containing x. The initial state of $\widehat{\Sigma}$ is also selected according to $\hat{x}_0 := \Pi_x(x_0)$ with x_0 being the initial state of Σ.

The proposed dynamical representation employs the map $\Pi_x : X \rightarrow \hat{X}$ that assigns to any $x \in X$, the representative point $\bar{x} \in \hat{X}$ of the corresponding partition set containing x satisfying the inequality:

$$\|\Pi_x(x) - x\| \leq \delta, \quad \forall x \in X, \tag{5}$$

where $\delta := \sup\{\|x - x'\|,\ x, x' \in \mathsf{X}_i,\ i = 1, 2, \ldots, n_x\}$ is the *state* discretization parameter.

Note that one can write the equivalent finite-MDP representation of $\widehat{\Sigma}$ in (4) as [39, Chapter 3.5]

$$\widehat{\Sigma} = (\hat{X}, \hat{U}, \hat{T}), \tag{6}$$

where

$$\hat{T}(x'|x, \nu) = T_{\mathsf{x}}(\Xi(x')|x, \nu), \quad \forall x, x' \in \hat{X}, \nu \in \hat{U},$$

and $\Xi : X \rightarrow 2^X$ is a map that assigns to any $x \in X$, the corresponding partition set it belongs to, i.e., $\Xi(x) = \mathsf{X}_i$ if $x \in \mathsf{X}_i$ for some $i = 1, 2, \ldots, n_x$. We employ this finite-MDP representation of (6) in Section IV.

The following theorem [5] shows the closeness between a continuous-space MDP Σ and its finite abstraction $\widehat{\Sigma}$ in a probabilistic setting.

Theorem 8. *Let $\Sigma = (X, U, \varsigma, f)$ be a continuous-space MDP and $\widehat{\Sigma} = (\hat{X}, \hat{U}, \varsigma, \hat{f})$ be its finite abstraction. For a given scLTL specification ϕ, and for any policy $\hat{\nu}(\cdot) \in \hat{\mathcal{U}}$ that preserves Markov property for the closed-loop $\widehat{\Sigma}$ (denoted by $\widehat{\Sigma}_{\hat{\nu}}$), the closeness between two systems can be acquired as*

$$|\mathbb{P}(\Sigma_{\hat{\nu}} \vDash \phi) - \mathbb{P}(\widehat{\Sigma}_{\hat{\nu}} \vDash \phi)| \leq \varepsilon, \quad \textit{with } \varepsilon := T\delta\mathscr{H}\mathscr{L}, \tag{7}$$

where T is the finite time horizon, δ is the state discretization parameter, $\mathscr{H}$ is the Lipschitz constant of the stochastic kernel, and $\mathscr{L}$ is the Lebesgue measure of the specification set. Moreover, optimal probabilities *of satisfying the specification over the two models are different with a distance of at most 2ε:*

$$\Big| \max_{\nu \in \bar{\Pi}_M} \mathbb{P}(\Sigma_\nu \vDash \phi) - \max_{\hat{\nu} \in \hat{\bar{\Pi}}_M} \mathbb{P}(\widehat{\Sigma}_{\hat{\nu}} \vDash \phi) \Big| \leq 2\varepsilon, \tag{8}$$

where $\bar{\Pi}_M$ and $\hat{\bar{\Pi}}_M$ are sets of Markov policies over Σ and $\widehat{\Sigma}$, respectively.

The error bound ε in (7) is obtained by characterizing $\mathbb{P}(\Sigma_{\hat{\nu}} \vDash \phi)$ recursively similar to dynamic programs (DP). This error is related to the approximation of the continuous kernel with a discrete one, hence the term $\delta\mathscr{H}$. There is also an integration over the specification set, thus $\mathscr{L}$ appears in ε. Finally, the errors contributed in every iteration of the DP are added, hence the horizon T.

Remark 9. *Note that in order to employ Theorem 8, one can first a-priori fix the desired threshold ε in* (7)*. According to the values of $\mathscr{H}$, $\mathscr{L}$, and T, one computes the required discretization parameter as $\delta = \frac{\varepsilon}{T\mathscr{H}\mathscr{L}}$. For instance in the case of a uniform quantizer, one can divide each dimension of the set X into intervals of size $\delta/\sqrt{n}$ with n being the dimension of the set.*

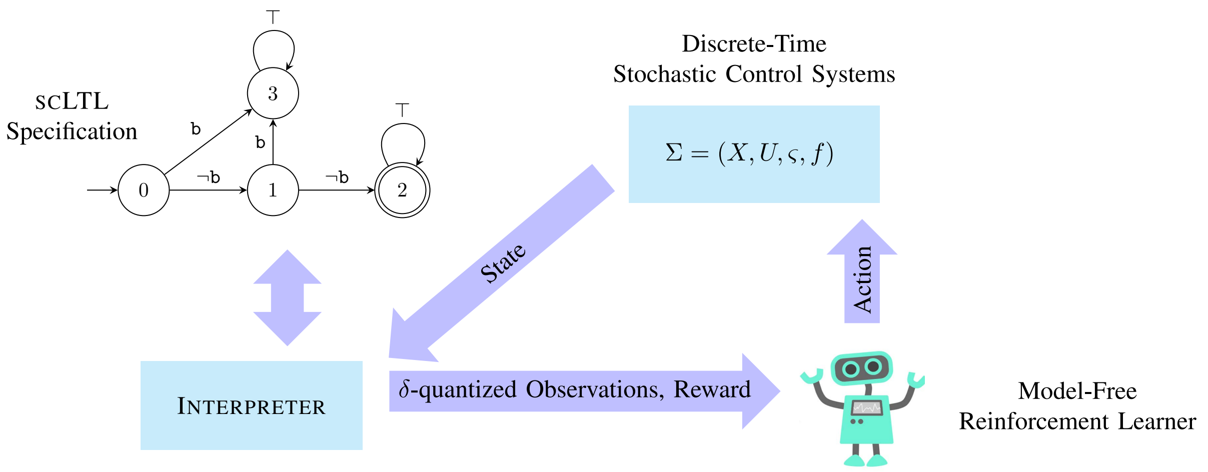
scLTL
Specification
⊤
3
b
b
0
¬b
1
¬b
2
⊤
Discrete-Time
Stochastic Control Systems
$\Sigma = (X, U, \varsigma, f)$
State
Action
INTERPRETER
δ-quantized Observations, Reward
Model-Free
Reinforcement Learner

Fig. 1. Model-free reinforcement learning is employed by DFA $\mathcal{A}_\phi$ corresponding to scLTL objective ϕ to provide scalar rewards by combining DFA $\mathcal{A}_\phi$ and a δ-quantized observation set of the continuous-space MDP Σ. In particular, the δ-quantized observation set of the continuous-space MDP Σ is used by an *interpreter* process to compute a run of $\mathcal{A}_\phi$. When the run of $\mathcal{A}_\phi$ reaches a final state, the interpreter gives the reinforcement learner a positive reward and the training episode terminates. Any converging reinforcement learning algorithm over such δ-quantized observation set is guaranteed to maximize the probability of satisfaction of the scLTL objective ϕ and converge to a 2ε-optimal strategy over the concrete dt-SCS Σ thanks to Theorem 8.

IV. Synthesis via Reinforcement Learning

In this section we sketch how we apply Theorem 8 to solve Problem 7 when conditional stochastic kernels are unknown. We begin by detailing the solution of finding optimal policies for scLTL properties in the case of known MDPs, and then we show how to exploit that to provide a reinforcement learning-based algorithm to synthesize an optimal policy.

A. Product MDP

It follows from Theorem 8 that one can construct a finite MDP $\widehat{\Sigma}$ from a continuous-space dt-SCS Σ with known conditional stochastic kernels such that the optimal probability of satisfaction of an scLTL specification ϕ for T steps in $\widehat{\Sigma}$ is no more than 2ε-worse than the optimal policy in Σ; see the definition of ε in Theorem 8. Hence, given a dt-SCS Σ with known conditional stochastic kernels, an scLTL property ϕ, and a time-horizon T, a 2ε-optimal policy to satisfy ϕ in T steps is computed using a suitable finite MDP with the corresponding δ as the state discretization parameter. This problem can be solved using the finite-horizon dynamic programming over the product of $\widehat{\Sigma}$ and the DFA $\mathcal{A}_\phi$ (cf. Definition 4 and the paragraph afterward) by giving a scalar reward to all transitions once a final state of $\mathcal{A}_\phi$ is reached.

Definition 10 (Product MDP)**.** *Given a finite MDP $\widehat{\Sigma} = (\hat{X}, \hat{U}, \hat{T})$ with initial state $\hat{x}_0 \in \hat{X}$, a labeling function $\mathsf{L} : X \to \Sigma_a$ (cf. Subsection II-B), and a DFA $\mathcal{A}_\phi = (Q, \Sigma_a, \mathsf{t}, q_0, F_a)$ capturing the scLTL specification ϕ, we define the* product MDP $\mathcal{M}_\star$ *as a finite MDP $(X_\star, U_\star, T_\star, x_\star, \rho_\star)$ where:*

- *$X_\star = \hat{X} \times Q$ is the set of states;*
- *$U_\star = \hat{U}$ is the set of actions;*
- *$T_\star : X_\star \times U_\star \times X_\star \to [0, 1]$ is the probabilistic transition function defined as*

$$T_\star((x,q),\nu,(x',q')) = \begin{cases} \hat{T}(x,\nu,x'), & \text{if } q' = \mathsf{t}(q, \mathsf{L}(x)),\\ 0, & \text{otherwise.}\end{cases}$$

- *$x_\star = (x_0, q_0)$ is the initial state; and*
- *$\rho_\star : X_\star \times U_\star \times X_\star \to \mathbb{N}$ is the reward function defined as:*

$$\rho_\star((x,q),\nu,(x',q')) = \begin{cases} 1, & \text{if } q \notin F \text{ and } q' \in F,\\ 0, & \text{otherwise.}\end{cases}$$

Recall that the DFA A_ϕ corresponding to an scLTL specification ϕ has the property that there is a unique accepting state and all out-going transitions from that state are self-loops. It follows that total optimal expected reward in the product is equal to the optimal probability of satisfying the specification.

Proposition 11 (Product Preserves Probability [40])**.** *An expected reward-optimal policy in $(X_\star, U_\star, T_\star, x_\star, \rho_\star)$ along with A_ϕ characterizes an optimal policy in $\widehat{\Sigma}$ to satisfy ϕ. The optimal expected total reward and an optimal policy can be computed in the polynomial time [41].*

B. Unknown Conditional Stochastic Kernels

When stochastic kernels are unknown, Theorem 8 still provides the correct probabilistic bound given a discretization parameter δ if the Lipschitz constant $\mathscr{H}$ is known. This observation enables us to employ reinforcement learning algorithms over the underlying discrete MDP without explicitly constructing the abstraction by simply restricting observations of the reinforcement learner to the closest representative point in the set of partitions (cf. Subsection III-A). We call such an underlying finite MDP a $\widehat{\Sigma}_\delta$ abstraction.

Model-free reinforcement learning can be employed under such observations by using the DFA $\mathcal{A}_\phi$ to provide scalar rewards as defined in Definition 10. The observations of the MDP are used by an *interpreter* process to compute a run of the DFA. When the DFA reaches a final state, the interpreter gives the reinforcement learner a positive reward and the training episode terminates. Since the product MDP $\mathcal{M}_\star$ is a finite MDP, from Proposition 11, it follows that any correct and convergent RL algorithm that maximizes this expected reward is guaranteed to converge to a policy that maximizes the probability of satisfaction of the scLTL objective. From Theorem 8 it then follows that any converging reinforcement learning algorithm [42], [43] over such finite observation set then converges to a 2ε-optimal policy over the concrete dt-SCS Σ thanks to Theorem 8. We summarize the proposed solution in the following theorem.

Theorem 12. *Let ϕ be an scLTL formula, $\varepsilon > 0$, and $\Sigma = (X, U, \varsigma, f)$ be a continuous-space MDP, where f and the distribution of ς are unknown but the Lipschitz constant $\mathscr{H}$ as discussed before is known. For a discretization parameter δ satisfying $T\delta\mathscr{H}\mathscr{L} \leq \varepsilon$, a convergent model-free reinforcement learning algorithm (e.g. Q-learning [43] or TD(λ) [42]) over $\widehat{\Sigma}_\delta$ with a reward function guided by the DFA $\mathcal{A}_\phi$, converges to a 2ε-optimal policy over Σ.*

C. Reward Shaping: Overcoming Sparse Rewards

Consider a finite MDP $\widehat{\Sigma} = (\hat{X}, \hat{U}, \hat{T})$, a co-safety automaton $\mathcal{A}_\phi = (Q, \Sigma_{\mathsf{a}}, \mathsf{t}, q_0, q_F)$, and their product MDP $\mathcal{M}_\star = (X_\star, U_\star, T_\star, x_\star, \rho_\star)$. Since the reward function $\rho_\star$ is sparse, it may not be effective in the reinforcement learning. For this reason, we introduce a "shaped" reward function ρ_κ (parameterized by a hyper-parameter κ) such that for suitable values of κ, optimal policies for ρ_κ are the same as optimal policies for $\rho_\star$, but unlike $\rho_\star$ the function ρ_κ is dense.

The function ρ_κ is defined based on the structure of co-safety automaton $\mathcal{A}_\phi$. Let $d(q)$ be the minimum distance of the state q to the unique accepting state q_F. Let $d_{\max} = 1 + \max_q\{d(q) : d(q) < \infty\}$. If there is no path from q to q_F, let $d(q)$ be equal to $d_{\max}$. We define the potential function $P : \mathbb{N} \to \mathbb{R}$ as the following:

$$P(d) = \begin{cases} \kappa \frac{d - d(q_0)}{1 - d_{\max}}, & \text{for } d > 0, \\ 1, & \text{for } d = 0, \end{cases}$$

where κ is a constant hyper-parameter. Note that the potential of the initial state $P(d(q_0)) = 0$ and the potential of the final state $P(d(q_F)) = 1$. Moreover, note that

$$P(1) - P(d_{\max}) = \kappa.$$

We define the "shaped" reward function $\rho_\kappa : \hat{X} \times \hat{U} \times \hat{X} \to \mathbb{R}$ as the difference between potentials of the destination and of the target states of transition of the automaton, i.e.,

$$\rho_\kappa((x, q), \nu, (x', q')) = P(d(q')) - P(d(q)).$$

Moreover, notice that for every run $r = (x_0, q_0), \nu_1, (x_1, q_1), \nu_2, \ldots, \nu_n, (x_n, q_n)$ of $\mathcal{M}_\star$, its accumulated reward is simply the potential difference between the last and the first states, i.e., $P(d(q_n)) - P(d(q_0))$.

Theorem 13 (Correctness of Reward Shaping)**.** *For every product MDP $\mathcal{M}_\star = (X_\star, U_\star, T_\star, x_\star, \rho_\star)$, there exists $\kappa_\star > 0$ such that for all $\kappa < \kappa_\star$ we have that the set of optimal expected reward policies for $\mathcal{M}_\star$ is the same as the set of optimal expected reward policies for $\mathcal{M}_\kappa = (X_\star, U_\star, T_\star, x_\star, \rho_\kappa)$.*

Proof. First we note that for the optimality, it is sufficient [44] to focus on positional strategies. Let μ_1 and μ_2 be two positional strategies such that the optimal probability of reaching the final state q_F for μ_1 is greater than that for μ_2. We write p_1 and p_2 for these probabilities and $p_1 > p_2$. Notice that these probabilities are equal to optimal expected reward with the $\rho_\star$ reward function.

We denote the expected total reward for policies μ_1 and μ_2 for the shaped reward function ρ_κ as s_1 and s_2, respectively. These rewards satisfy the following inequalities:

$$\begin{aligned} s_1 &\geq p_1(P(0) - P(d(q_0))) + (1 - p_1)(P(d_{\max}) - P(d(q_0)), \\ s_2 &\leq p_2(P(0) - P(d(q_0))) + (1 - p_2)(P(1) - P(d(q_0))). \end{aligned}$$

It can be verified that if $\kappa < p_1 - p_2$, then $s_1 > s_2$. Therefore, if μ_1 is an optimal positional strategy, and μ_2 is one of the next best positional strategies, choosing $\kappa_* < p_1 - p_2$ guarantees that an optimal strategy in $\mathcal{M}_\kappa$ is also optimal for $\mathcal{M}_\star$. $\square$

Theorem 13 demonstrates one way to shape rewards such that the optimal policy remains unaffected while making the rewards less sparse. Along similar lines, one can construct a variety of potential functions and corresponding shaped rewards with similar correctness properties. Of course, the reward shaping schema presented here is no silver bullet: we expect the performance of different potential functions to be incomparable along a carefully chosen ensemble of MDPs. Since rewards are shaped without any knowledge of the underlying MDP, there may be MDPs where un-shaped rewards may work as well or even better than a given shaped reward. We envisage that the ability to combine several competing ways to shape reward may work better in practice. While sparse rewards may be sufficient for simpler learning tasks, we demonstrate that shaped rewards such as the one provided here are crucial for larger case studies such as the BMW case-study reported in the next section.

V. Case Studies

Before illustrating our results via some experiments, we elaborate on the dimension dependency in our proposed RL techniques compared to the abstraction-based ones. Assuming a uniform quantizer, the finite MDP constructed in Subsection III-A is a matrix with a dimension of $(n_x \times n_u) \times n_x$, where n_u is the cardinality of the finite input set U. Computing this matrix is one of the bottlenecks in abstraction-based approaches since an n-dimensional integration has to be done numerically for each entries of this matrix. Moreover, n_x (i.e., the cardinality of the finite state set) grows exponentially with the dimension n. Once this matrix is computed, it is employed for the dynamic programming on a vector of the size $(n_x \times n_u)$. This is a second bottleneck of the process. On the other hand, by employing the proposed RL approach, the curse of dimensionality reduces to only *learning* the vectors of size $(n_x \times n_u)$ without having to compute the full matrix. Moreover, the abstraction-based techniques need to precisely know the probabilistic evolution of states in models, whereas in this work we only need to know the Lipschitz constant $\mathscr{H}$.

Concerning the trade-off between iteration count, discretization size, and performance, we should mention that by decreasing the discretization parameter, the closeness error in Theorem 8 is reduced. On the other hand, one needs more training episodes as the size of the problem increases. Note that in our proposed setting, we do not need to compute transition probabilities $\hat{T}$ for finite MDPs $\widehat{\Sigma}$, since we directly learn the value functions using RL.

To demonstrate the effectiveness of the proposed results, we first apply our proposed approaches to two physical benchmarks including regulation of room temperature and control of road traffic. We then apply our algorithms to a nonlinear model of a BMW 320i car by synthesizing a controller enforcing a reach-while-avoid specification. The first two case studies are intentionally chosen to be small such that we can compare (cf. Table I) probabilities of satisfaction p_r estimated by RL with the optimal probabilities p_* computed using the dynamic programming when the exact dynamics are known.

A. Room Temperature Control

Here, we apply our results to the temperature regulation of a room equipped with a heater. The model of this case study is adapted from [45] by including stochasticity in the model as an additive noise. The evolution of the temperature can be described by the following dt-SCS:

$$\Sigma : x(k+1) = (1 - 2\eta - \beta - \gamma\nu(k))x(k) + \gamma T_h \nu(k) + \beta T_e + 0.3162\varsigma(k),$$

where $\eta = 0$, $\beta = 0.022$, and $\gamma = 0.05$ are conduction factors respectively between this room and the other rooms in a network, between the external environment and the room, and between the heater and the room. Moreover, $x(k)$ and $\nu(k)$ are taking values in $[19, 21]$ and a finite input set $\{0.03, 0.09, 0.15, 0.21, 0.27, 0.33, 0.39, 0.45, 0.51, 0.57\}$, respectively. The parameter $T_e = -1\,^\circ C$ is the outside temperature, and $T_h = 50\,^\circ C$ is the heater temperature.

The main goal is to synthesize a controller for Σ using our main results in Theorem 12 such that the controller maintains the temperature of the room in the safe set $[19, 21]$.

B. Road Traffic Control

We also apply our results to a road traffic control containing a cell with 2 entries and 1 way out, as schematically depicted in Figure 2. The model of this case study is taken from [46]; stochasticity is included in the model as the additive noise. One of the entries of the cell is controlled by a traffic light, denoted by $\nu = \{0, 1\}$, that enables (green light) or not (red light) the vehicles to pass. In this model, the length of a cell

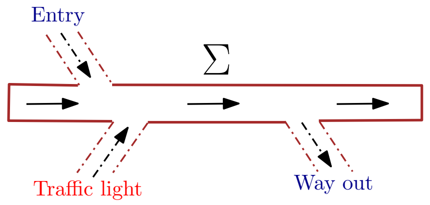

Fig. 2. Model of a road traffic control with the length of 500 meters, 1 way out, and 2 entries, one of which is controlled by a traffic light.

is 0.5 kilometers ([km]), and the flow speed of the vehicles is 100 kilometers per hour ([km/h]). Moreover, during the time interval $\tau = 6.48$ seconds, it is assumed that 6 vehicles pass the entry controlled by the traffic light, 3 vehicles go into the entry of the cell, and one quarter of vehicles goes out on the exit of the cell (the ratio denoted by q). We want to observe the density of traffic x. The model of the system Σ is described by:

$$\Sigma : x(k+1) = (1 - \frac{\tau v}{l} - q)x(k) + 6\nu(k) + 1.9494\varsigma(k) + 3,$$

where l and v are the length of the cell and the flow speed of vehicles, respectively. We synthesize a controller for Σ using our main results in Theorem 12 such that the density of the traffic is lower than 20 vehicles.

C. Experiments

Table I shows a comparison of Q-learning to the computed optimal probabilities for the room temperature and road traffic examples. For each model, five different discretization steps (δ) are considered and for each value of δ probabilities of satisfaction of the safety objectives are reported in the columns labeled p_r. These probabilities are Q-values of the initial state of the finite-state MDP for the policy computed by Q-learning after 10^6 episodes. The objective is to keep the system safe for at least 10 steps. For the comparison, the optimal probability p_* for a time-dependent policy is reported assuming that we know the exact dynamics for these two examples. Note that we compute p_* using the dynamic programming over constructed finite MDPs as proposed in Subsection III-A. The optimal probability p_* reported in Table I corresponds to the same initial condition that is utilized in the learning process. The optimal probability for the original *continuous-space* MDP is always within an interval $[p_l, p_h]$ centered at p_* and with a radius ε as reported in Table I. One can readily see from Table I that as the discretization parameter δ decreases, the size of this interval shrinks, which implies that the optimal probability for the original *continuous-space* MDP converges to p_*. While finer abstractions give better theoretical guarantees, for a fixed number of episodes it is easier to learn good strategies for coarser abstractions. This is reflected in Table I, where the values of p_r do not necessarily get better with smaller values of δ. However, by increasing the number of episodes, strategies converge toward the optimal one, as illustrated in Figure 3, which visualizes room temperature control strategies computed by Q-learning after different numbers of episodes. Note that in Table I, the error bound ε exceeds one for $\delta \geq 0.05$ in

TABLE I
Q-Learning Results for Room Temperature and Road Traffic Examples.

	Room					Traffic				
δ	p_r	p_*	ε	p_l	p_h	p_r	p_*	ε	p_l	p_h
0.01	0.9698	0.9753	0.2468	0.7285	1.0	0.9856	0.9995	0.0160	0.9835	1.0
0.02	0.9745	0.9753	0.4936	0.4817	1.0	0.9975	0.9995	0.0319	0.9676	1.0
0.05	0.9543	0.9753	1.2339	0.0000	1.0	0.9993	0.9995	0.0798	0.9197	1.0
0.1	0.9779	0.9754	2.4678	0.0000	1.0	0.9999	0.9995	0.1596	0.8399	1.0
0.2	0.9732	0.9743	4.9357	0.0000	1.0	0.9999	0.9995	0.3193	0.6802	1.0

the room temperate control example, which is not a useful probability bound for the *continuous-space* MDP. However, we prefer to report the corresponding values of p_r and p_* so that they can still be compared.

D. 7-Dimensional Autonomous Vehicle

The previous case-studies are representative of what can be solved by discretization and tabular methods like Q-learning. Relaxing those constraints, we were able to apply deep deterministic policy gradient (DDPG) [20] to a 7-dimensional *nonlinear* model of a BMW 320i car [47] to synthesize a reach-while-avoid controller. Though convergence guarantees are not available for DDPG and most RL algorithms with nonlinear function approximations, breakthroughs in this direction (e.g., SBEED in [26]) will expand the applicability of our results to more complex safety-critical applications.

The model of this case study is borrowed from [47, Section 5.1] by discretizing the dynamics in time and including a stochasticity inside the dynamics as additive noises. We are interested in an autonomous operation of the vehicle on a highway. Consider a situation on a two-lane highway when an accident suddenly happens on the same lane on which our vehicle is traveling. The vehicle's controller should find a safe maneuver to avoid the crash with the next-appearing obstacle.

Figure 4 shows the simulation from 100 samples with varying initial positions and initial heading velocities (16–18 m/s) for the learned controller. We employed potential-based reward shaping to speed-up learning in this case study from 10K episodes (no success) to under 5K episodes (for a convincing learning, see Figure 4).

VI. Conclusion

We studied the problem of finding policies for systems that can be modeled as continuous-space MDPs but with unknown dynamics. The goal of the policy is to maximize the probability that the system satisfies a complex property expressed as a fragment of linear temporal logic formulae. Our approach replaces the unknown system with a finite MDP without explicitly constructing it. Since transition probabilities of the finite MDP are unknown, we utilize the reinforcement learning (RL) to find a policy and apply it to the original continuous-space MDP. We show that any converging reinforcement learning algorithm [42], [43] over such finite observation MDP converges to a 2ε-optimal strategy over the concrete continuous-space MDP with unknown dynamics (only an upper bound on the Lipschitz constant is known), where ε is defined a-priori and can be controlled. We applied our approach to multiple case studies. The results are promising and demonstrate that by employing an automata-theoretic reward shaping, the learning algorithm enlarges the class of systems over which we can perform the formal synthesis.

References

[1] C. Baier and J.-P. Katoen, *Principles of Model Checking*. MIT Press, 2008.

[2] A. Pnueli, "The temporal logic of programs," in *Symposium on Foundations of Computer Science*, 1977, pp. 46–57.

[3] A. Abate, M. Prandini, J. Lygeros, and S. Sastry, "Probabilistic reachability and safety for controlled discrete time stochastic hybrid systems," *Automatica*, vol. 44, no. 11, pp. 2724–2734, 2008.

[4] I. Tkachev and A. Abate, "On infinite-horizon probabilistic properties and stochastic bisimulation functions," in *Proceedings of the 50th IEEE Conference on Decision and Control*, 2011, pp. 526–531.

[5] S. Soudjani and A. Abate, "Adaptive and sequential gridding procedures for the abstraction and verification of stochastic processes," *SIAM J. on Applied Dynamical Systems*, vol. 12, no. 2, pp. 921–956, 2013.

[6] R. Majumdar, K. Mallik, and S. Soudjani, "Symbolic controller synthesis for Büchi specifications on stochastic systems," *Hybrid Systems: Computation and Control, arXiv:1910.12137*, 2020.

[7] S. Prajna, A. Jadbabaie, and G. J. Pappas, "A framework for worst-case and stochastic safety verification using barrier certificates," *IEEE Trans. on Automatic Control*, vol. 52, no. 8, pp. 1415–1428, 2007.

[8] P. Jagtap, S. Soudjani, and M. Zamani, "Formal synthesis of stochastic systems via control barrier certificates," *arXiv: 1905.04585*, 2019.

[9] K. G. Larsen and A. Skou, "Bisimulation through probabilistic testing," *Information and Computation*, vol. 94, no. 1, pp. 1–28, 1991.

[10] R. Segala and N. Lynch, "Probabilistic simulations for probabilistic processes," *Nordic J. of Computing*, vol. 2, no. 2, pp. 250–273, 1995.

[11] J. Desharnais, F. Laviolette, and M. Tracol, "Approximate analysis of probabilistic processes: Logic, simulation and games," in *Proceedings of the 5th International Conference on Quantitative Evaluation of System*, 2008, pp. 264–273.

[12] A. A. Julius and G. J. Pappas, "Approximations of stochastic hybrid systems," *IEEE Trans. on Automatic Control*, vol. 54, no. 6, pp. 1193–1203, 2009.

[13] A. Abate, M. Kwiatkowska, G. Norman, and D. Parker, "Probabilistic model checking of labelled Markov processes via finite approximate bisimulations," in *Horizons of the Mind. A Tribute to Prakash Panangaden*, 2014, pp. 40–58.

[14] S. Haesaert, S. Soudjani, and A. Abate, "Verification of general Markov decision processes by approximate similarity relations and policy refinement," *SIAM J. on Control and Optimization*, vol. 55, no. 4, pp. 2333–2367, 2017.

[15] S. Haesaert and S. Soudjani, "Robust dynamic programming for temporal logic control of stochastic systems," *TAC, arXiv/1811.11445*, 2020.

[16] R. S. Sutton and A. G. Barto, *Reinforcement Learnging: An Introduction*, 2nd ed. MIT Press, 2018.

[17] A. L. Strehl, L. Li, E. Wiewiora, J. Langford, and M. L. Littman, "PAC model-free reinforcement learning," in *International Conference on Machine Learning*, 2006, pp. 881–888.

[18] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Machine Learning*, vol. 3, no. 1, pp. 9–44, 1988.

[19] C. J. C. H. Watkins, "Learning from delayed rewards," Ph.D. dissertation, King's College, Cambridge, 1989.

[20] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *CoRR*, vol. abs/1509.02971, 2015.

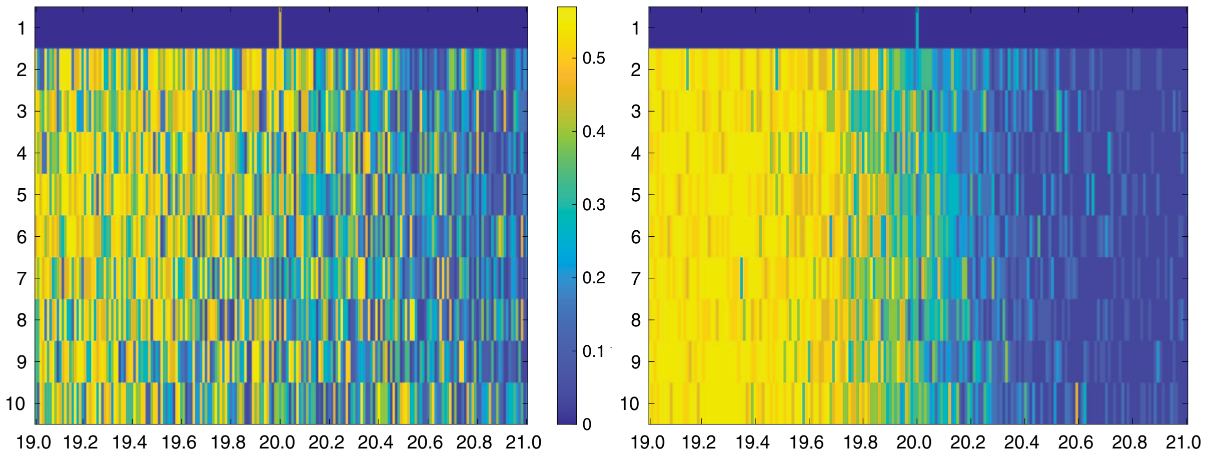

Fig. 3. Room temperature control: A heatmap visualization of strategies learned via Reinforcement Learning after 10^5 episodes (left) and after $8 \cdot 10^6$ episodes (right). The X axis represents the room temperature in ℃, while the Y axis represents time steps $1 \leq k \leq 10$. The action suggested by the strategy is in the finite input set $\{0.03, 0.09, 0.15, 0.21, 0.27, 0.33, 0.39, 0.45, 0.51, 0.57\}$ and is color-coded according to the map shown in the middle: Bright yellow and deep blue represent maximum and minimum heat. In the first step, strategies are only defined for the initial state; this causes the blue bands at the top.

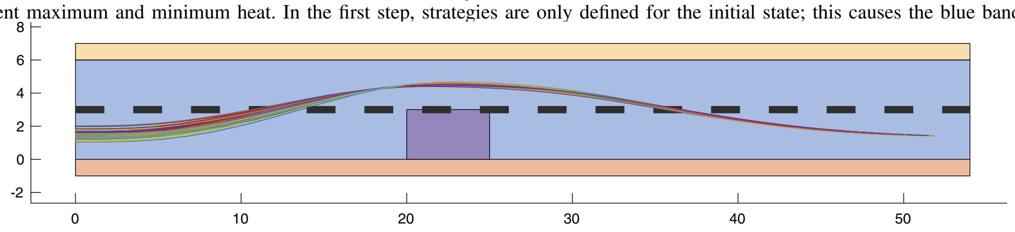

Fig. 4. Trajectories of 100 simulations of the RL-synthesized controller for a 7-dimensional model of a BMW 320i car trained using DDPG. The road is 6 meter wide and 50 meter long, and the length of the car is .4 508 meters and its width is .1 610 meters.

[21] M. Riedmiller, "Neural fitted Q iteration – First experiences with a data efficient neural reinforcement learning method," in *Machine Learning: ECML 2005*, 2005, pp. 317–328.

[22] G. Tesauro, "Temporal difference learning and TD-Gammon," *Commun. ACM*, vol. 38, no. 3, pp. 58–68, 1995.

[23] V. Mnih *et al.*, "Human-level control through reinforcement learning," *Nature*, vol. 518, pp. 529–533, 2015.

[24] D. Silver *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, pp. 484–489, 2016.

[25] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 1334–1373, Jan. 2016.

[26] B. Dai, A. Shaw, L. Li, L. Xiao, N. He, Z. Liu, J. Chen, and L. Song, "SBEED: Convergent reinforcement learning with nonlinear function approximation," *arXiv:1712.10285*, 2017.

[27] A. Y. Ng, D. Harada, and S. J. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *International Conference on Machine Learning*, 1999, pp. 278–287.

[28] M. Hasanbeig, A. Abate, and D. Kroening, "Certified reinforcement learning with logic guidance," *arXiv:1902.00778*, 2019.

[29] M. Hasanbeig, Y. Kantaros, A. Abate, D. Kroening, G. J. Pappas, and I. Lee, "Reinforcement learning for temporal logic control synthesis with probabilistic satisfaction guarantees," *arXiv:1909.05304*, 2019.

[30] L. Z. Yuan, M. Hasanbeig, A. Abate, and D. Kroening, "Modular deep reinforcement learning with temporal logic specifications," *arXiv:1909.11591*, 2019.

[31] R. Munos, "A convergent reinforcement learning algorithm in the continuous case: the finite-element reinforcement learning," in *International Conference on Machine Learning*, January 1996.

[32] D. Silver *et al.*, "Deterministic policy gradient algorithms," in *Proceedings of the 31st International Conference on International Conference on Machine Learning*, 2014, pp. 387–395.

[33] D. P. Bertsekas and S. E. Shreve, *Stochastic Optimal Control: The Discrete-Time Case*. Athena Scientific, 1996.

[34] O. Kupferman and M. Y. Vardi, "Model checking of safety properties," *Formal Methods in System Design*, vol. 19, no. 3, pp. 291–314, 2001.

[35] J. Desharnais, F. Laviolette, and M. Tracol, "Approximate analysis of probabilistic processes: logic, simulation and games," in *Proceedings of the International Conference on Quantitative Evaluation of SysTems (QEST 08)*, 2008, pp. 264–273.

[36] D. W. Scott, *Multivariate Density Estimation. Theory, Practice, and Visualization*. Wiley, 1992.

[37] M. P. Holmes, A. G. Gray, and C. L. Isbell, "Fast nonparametric conditional density estimation," *arXiv: 1206.5278*, 2012.

[38] O. Kallenberg, *Foundations of modern probability*. Springer-Verlag, New York, 1997.

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

[40] C. Courcoubetis and M. Yannakakis, "The complexity of probabilistic verification," *J. ACM*, vol. 42, no. 4, pp. 857–907, 1995.

[41] C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of Markov decision processes," *Mathematics of operations research*, vol. 12, no. 3, pp. 441–450, 1987.

[42] T. Jaakkola, M. Jordan, and S. P. Singh, "Convergence of stochastic iterative dynamic programming algorithms," in *Advances in neural information processing systems*, 1994, pp. 703–710.

[43] V. S. Borkar and S. P. Meyn, "The ODE method for convergence of stochastic approximation and reinforcement learning," *SIAM J. on Control and Optimization*, vol. 38, no. 2, pp. 447–469, 2000.

[44] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. New York: John Wiley & Sons, 1994.

[45] P. J. Meyer, A. Girard, and E. Witrant, "Compositional abstraction and safety synthesis using overlapping symbolic models," *IEEE Trans. on Automatic Control*, vol. 63, no. 6, pp. 1835–1841, 2018.

[46] E. Le Corronc, A. Girard, and G. Goessler, "Mode sequences as symbolic states in abstractions of incrementally stable switched systems," in *Conference on Decision and Control*, 2013, pp. 3225–3230.

[47] M. Althof, "Commonroad: Vehicle models (version 2018a). Tech. rep." Technical University of Munich, 85748 Garching, Germany (October 2018), https://commonroad.in.tum.de, 2019.