Sampling-based Stochastic Optimal Control with Metric Interval Temporal Logic Specifications

Felipe J. Montana, Jun Liu and Tony J. Dodd

Abstract— **This paper describes a method to find optimal policies for stochastic dynamic systems that maximise the probability of satisfying real-time properties. The method consists of two phases. In the first phase, a coarse abstraction of the original system is created. In each region of the abstraction, a sampling-based algorithm is utilised to compute local policies that allow the system to move between regions. Then, in the second phase, the selection of a policy in each region is obtained by solving a reachability problem on the Cartesian product between the abstraction and a timed automaton representing a real-time specification given as a metric interval temporal logic formula. In contrast to current methods that require a fine abstraction, the proposed method achieves computational tractability by modelling the coarse abstraction of the system as a bounded-parameter Markov decision process (BMDP). Moreover, once the BMDP is created, this can be reused for new specifications assuming the same stochastic system and workspace. The method is demonstrated with an autonomous driving example.**

I. INTRODUCTION

Motion planning based on high-level temporal specifications has become an important area of research. During the last few years, several methods have been developed for deterministic, e.g., [1][2][3], non-deterministic, e.g., [4][5][6] and stochastic systems, e.g., [7][8]. These methods specify properties using temporal logics such as linear temporal logic (LTL) and computation tree logic (CTL). Although useful missions can be stated using these logics, they are limited to qualitative specifications. In other words, only the order of events can be expressed. To solve this limitation, methods using real-time logics such as metric temporal logic (MTL) [9] have been proposed. In [10], the authors develop a method to abstract a continuous system preserving MTL properties and propose a technique to transform MTL formulae to LTL formulae allowing to apply existing methods for discrete systems to find a solution. Using a mathematical programming-based approach, a robust control is obtained for non-deterministic systems based on signal temporal logic (STL) in [11]. By encoding specifications as constraints of a mixed integer linear programming (MILP), the possible computationally expensive process of abstraction is avoided.

For stochastic dynamics, the authors in [12] compute an optimal policy with respect to the probability of satisfying a metric interval temporal logic (MITL) specification. The solution is found in the product operation between a timed automaton representing a desired specification and a discrete abstraction that approximates a continuous-time stochastic system. They prove that the optimality of the computed policy depends on the level of granularity in the state and time space abstraction. The limitation of the approach is the scalability: as the abstraction gets finer, the number of states becomes intractable.

As in the aforementioned work, because of discretisation, the complexity scales exponentially with the dimension of the state space in several methods presented in the literature. To mitigate this problem, sampling-based algorithms have been proposed, e.g., [13][14][15][16]. Although these methods partially solve the problem of scalability, they do not address the problem for systems with stochastic dynamics or real-time specifications. An exception is [17], where a solution is presented for stochastic systems but real-time specifications are not considered. The method uses a sampling-based algorithm to compute local policies within discrete regions of the state space to reach local goals. A policy is then selected by solving a product bounded-parameter Markov decision process (BMDP) of the discretisation and an automaton representing a co-safe LTL formula.

In this paper we extend the work in [17] by finding optimal policies for stochastic system based on MITL specifications. Although, as described above, this problem has been addressed in [12], the solution is limited by its scalability. In contrast, our method achieves computational tractability by dividing the solution into two phases. During the first phase, the system is coarsely discretised in distinct regions. Then, local policies are computed within each discrete region to drive the system to adjacent regions. In the second phase, similar to [12], a Cartesian product between the abstraction and a timed automaton is used to compute an optimal global policy that selects local policies in each region. This global policy is optimal in the sense that the probability of satisfying a real-time specification is maximised. The main contribution of this paper is a method that utilises a coarse abstraction to reduce complexity and allows a fast computation of policies. Once local policies are computed in the abstraction, only the second phase has to be repeated for new MITL specifications. Furthermore, the dynamics of the system under consideration only has an impact in the first phase. Hence, the complexity of recomputing new policies only depends on the number of regions created during the abstraction.

This work was supported, in part, by the Mexican National Council of Science and Technology (CONACyT).

Felipe J. Montana and Tony J. Dodd are with the Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield S1 3JD, UK. E-mail: fjmontanagonzalez1,t.j.dodd@sheffield.ac.uk.
Jun Liu is with the Department of Applied Mathematics, University of Waterloo, Waterloo N2L 3G1, Canada. E-mail: j.liu@uwaterloo.ca

II. PRELIMINARIES AND PROBLEM DEFINITION

Notation: For a set Q, let $|Q|$, 2^Q, Q^o and ∂Q denote its cardinality, power set, interior and boundary, respectively. We use $\mathbb{Z}_+$ to denote the set of non-negative integers and $\mathbb{R}_+$ for non-negative real numbers. For $n,m \in \mathbb{Z}_+$, $\mathbb{R}^n$ and $\mathbb{R}^{n\times m}$ are the set of column vectors and matrices with n and $n \times m$ real entries.

A. System Model

This paper focuses on stochastic dynamic systems, called controlled diffusions, that evolve according to the stochastic differential equation:

$$dx(t) = f(x(t),u(t))dt + G(x(t))dw(t), \tag{1}$$

where $x \in X \subset \mathbb{R}^{d_x}$ is the system state and $u \in U \subset \mathbb{R}^{d_u}$ is the control input. $\mathbb{R}^{d_x}$ and $\mathbb{R}^{d_u}$ are the d_x-dimensional and d_u-dimensional Euclidean space, respectively. $f : X \times U \to \mathbb{R}^{d_x}$ and $G : X \to \mathbb{R}^{d_x \times d_x}$ are bounded continuous functions; and $w(\cdot)$ is a d_w-dimensional Wiener process on a probability space $(\Omega, \mathscr{F}, \mathscr{P})$. The matrix $G(\cdot)$ is assumed to have full rank and the control $u(\cdot)$ is admissible with respect to $w(\cdot)$[18].

B. Bounded-parameter Markov Decision Process

A bounded-parameter Markov decision process (BMDP) [19] is used to abstract the motion of the continuous stochastic system, see Section III. An BMDP is a tuple $\mathscr{B} = (S, A, \hat{P}, \check{P})$, where:

- S is a finite set of states,
- A is a finite set of actions,
- $\hat{P}(\cdot|\cdot,\cdot) : S \times S \times A \to [0,1]$ is the upper bound of the probability of transitioning to the state s_j from the state s_i under action $a \in A$,
- $\check{P}(\cdot|\cdot,\cdot) : S \times S \times A \to [0,1]$ is the lower bound of the probability of transitioning to the state s_j from the state s_i under action $a \in A$.

For all states $s \in S$ and any action $a \in A$, the probability functions $\hat{P}$ and $\check{P}$ satisfy the following conditions:

$$0 \le \check{P}(\cdot|s,a) \le \hat{P}(\cdot|s,a) \le 1, \tag{2}$$

$$0 \le \sum_{j=1}^{|S|} \check{P}(s_j|s_i,a) \le 1 \le \sum_{j=1}^{|S|} \hat{P}(s_j|s_i,a). \tag{3}$$

C. Metric Interval Temporal Logic

We use metric interval temporal logic (MITL) [20] to express system properties. These properties are represented by a set Π of atomic propositions that indicate whether a property is true or false. A labelling function $L_X : X \to 2^\Pi$ maps each state $x \in X$ to the set Π.

Syntax: The syntax of MITL over the set Π is defined as follows:

$$\varphi := \pi \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \mathscr{U}_I \varphi_2,$$

where $\pi \in \Pi$, $\neg$, $\vee$, $\wedge$ and $\mathscr{U}$ represent the operators *negation*, *disjunction*, *conjunction* and *until*, respectively; and I is a convex subset of $\mathbb{R}_+$ of the form: (a,b), $(a,b]$, $[a,b)$ or $[a,b]$, where $a,b \in \mathbb{Z}_+$ and $a < b$. The temporal operators *eventually* and *always* are defined as $\Diamond_I \pi = True \mathscr{U}_I \pi$ and $\Box_I \pi = \neg\Diamond_I \neg\pi$, respectively. Only formulae in positive normal form (PNF), where negations only occur in front of atomic propositions [21], are considered.

Continuous semantics: A signal function $\xi : [0,\infty] \to X$ is used to interpret MITL formulae. Given the function ξ, the satisfaction relation $\models$ and an MITL formula φ, we define $\xi(t) \models \varphi$ inductively as follows:

$$\xi(t) \models \pi \text{ iff } \pi \in L_X(\xi(t)),$$
$$\xi(t) \models \varphi_1 \wedge \varphi_2 \text{ iff } \xi(t) \models \varphi_1 \text{ and } \xi(t) \models \varphi_2,$$
$$\xi(t) \models \varphi_1 \vee \varphi_2 \text{ iff } \xi(t) \models \varphi_1 \text{ or } \xi(t) \models \varphi_2,$$
$$\xi(t) \models \varphi_1 \mathscr{U}_I \varphi_2 \text{ iff } \exists t' \in I : \xi(t') \models \varphi_2 \text{ and } \xi(t) \models \varphi_1,\ \forall t \in [a,t').$$

D. Timed Automaton

We limit system specifications to those MITL formulae that can be converted into deterministic timed automaton (DTA). Let $C = \{c_1, c_2, \ldots, c_{d_c}\}$ be a finite set of real-valued clocks. Similar to [12], we define a clock vector $\mathbf{C} \in \mathbb{R}^{d_c}$ with entries equal to the value of each clock. The i-th entry of the clock vector is denoted by $\mathbf{C}[i]$. For the set C, $\Lambda(C)$ is a set of constraints defined as:

$$\lambda := c \le k \mid c \ge k \mid \neg\lambda \mid \lambda_1 \wedge \lambda_2,$$

where λ is a clock constraint in $\Lambda(C)$, $c \in C$ and $k \in \mathbb{Z}_+$.

A deterministic timed automaton [22] is a tuple $\mathscr{T} = (\Sigma, Q, q_0, Q_F, C, \Lambda(C), \to)$, where:

- $\Sigma = 2^\Pi$ is a finite alphabet,
- Q is a finite set of states,
- $q_0 \in Q$ is an initial state,
- $Q_F \subset Q$ is a set of accepting states,
- C is a finite set of clocks,
- $\Lambda(C)$ is a set of clock constraints,
- $\to \subseteq Q \times Q \times \Sigma \times 2^C \times \Lambda(C)$ is a transition relation.

A configuration of $\mathscr{T}$ is defined by a pair $(q, \mathbf{C})$, where $q \in Q$ is a state and $\mathbf{C}$ is the clock vector defined above. We write $(q,\mathbf{C}) \xrightarrow{\pi,t} (q',\mathbf{C}')$ to represent a transition from configuration $(q,\mathbf{C})$ to configuration $(q',\mathbf{C}')$ on input $\pi \in \Pi$ after t units of time, where $\mathbf{C}'[i] = \mathbf{C}[i] + t$ for $i \in \{1,\ldots,d_c\}$. After the transition, the set of clocks $\delta \subseteq C$ are reset to zero, i.e., $C'[i] = 0$ if $c_i \in \delta$.

In order to describe the behaviour of the system over time, let $\eta = \{\eta_i\}_{i=0}^{\infty}$, where $\eta_i \in \mathbb{R}_+$, be an infinite time sequence that satisfies: $\eta_i < \eta_{i+1}$ for all $i \ge 0$ and for all $t \in \mathbb{R}_+$, there is some i such that $\eta_i > t$. Given the alphabet Σ and the time sequence η, a timed word is defined by the pair (σ,η), where $\sigma = \sigma_1\sigma_2\ldots$ is a word over Σ. Since the alphabet Σ is formed by the set of atomic propositions Π, a timed word gives the time at which system properties occurs, i.e., $\sigma_i = L_X(\xi(\eta_i))$.

Let $\Delta\eta_i = \eta_i - \eta_{i-1}$. A run in $\mathscr{T}$ on a timed word (σ,η) is an infinite sequence $(q_0,\mathbf{C}_0) \xrightarrow{\sigma_1,\Delta\eta_1} (q_1,\mathbf{C}_1) \xrightarrow{\sigma_2,\Delta\eta_2} \ldots,$

where $\mathbf{C}_0$ is the clock vector with all entries equal to zero and for all $i \geq 1$, $\mathbf{C}_i[j] = \mathbf{C}_{i-1}[j] + \Delta\eta_i$ if $c_j \notin \delta$, $\mathbf{C}_i[j] = 0$ if $c_j \in \delta$ and in each transition $\mathbf{C}_i$ satisfies the clock constraint $\lambda_i \in \Lambda(C)$. A timed word (σ, η) is accepted by a timed automaton $\mathscr{T}$ if a state $q \in Q_F$ is visited in the run produced by (σ, η).

E. Problem Formulation

We say that the system satisfies the specification φ in the discrete semantics if the timed word (σ, η), describing the behaviour of a sample path of the system, is accepted by the timed automaton $\mathscr{T}_\varphi$, where $\mathscr{T}_\varphi$ is the timed automaton that accepts runs satisfying φ. On the other hand, if $\{\Delta\eta_i \to 0, i \geq 1\}$ and (σ, η) is accepted by $\mathscr{T}_\varphi$, we say that the system satisfies the specification φ in the continuous semantics. Given these definitions, the problem addressed can be formally defined.

Problem definition: Given a stochastic dynamic system of the form (1) and an MITL formula φ, compute a control policy $\mu : X \to U$ such that the probability of satisfying φ in the continuous semantics is maximised.

III. Solution

This section presents a solution for finding a policy that maximises the probability of a stochastic system of the form (1) satisfying a specification given as an MITL formula. To reduce computational complexity, the proposed method is divided into two phases. First, the system is coarsely discretised in distinct regions. In each region, local policies that drive the system from one region to another are computed, Figure 1. This calculation relies on a sampling-based algorithm called iMDP [23], which approximates the model in (1) using the Markov chain method [18]. Since the probability of transitioning from one region to another depends on the initial state of the stochastic system within the region, a range of probabilities is required to represent transitions between regions. To model this range, a BMDP [19] is utilised.

In the second phase, the MITL formula is converted into a DTA. Then, the original problem is reformulated as a reachability problem in the product automaton between the BMDP and the DTA. The solution of the reachability problem yields an optimal global policy that is used to select a local policy in each region. In contrast to [12], the size of the product automaton, where the solution is found, only depends on the number of regions and not the dynamics of the system. Moreover, note that for new specifications, only the second phase of the method has to be solved. Therefore, the main benefit of the method is the ability to fast recalculate global policies for new specifications. The rest of the section explains the computation of local polices (Section III.A), the construction of the BMDP and product automaton (Section III.B and III.C) and the solution to the reachability problem (Section III.D).

A. Discretisation and Local Policies

In this work, the workspace Γ is decomposed by a Delaunay triangulation. Nevertheless, this process can be performed by any other partitioning method. Since the workspace is a projection of X onto $\mathbb{R}^{d_\Gamma}$, where d_Γ is the dimension of the workspace Γ, this decomposition induces a discretisation in X. The lower dimensionality of the decomposed workspace Γ avoids the exponential computational cost of decomposing the state space. Let $S = \{s_1, \ldots, s_{d_S}\}$ be the set of regions obtained after the decomposition. In each region $\{s_i, i \in \{1, \ldots, d_s\}\}$, a local policy is computed, for each adjacent region, using the iMDP algorithm. The local policies are used to drive the system from region s_i to any of its adjacent regions, Figure 1. The computation of local policies within the region s_i is now presented.

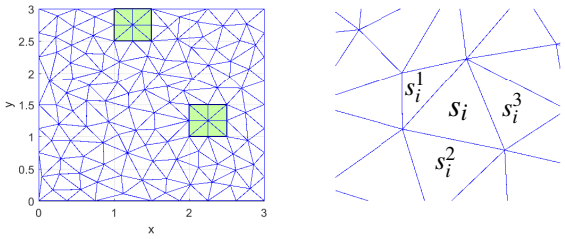

Fig. 1: Discretisation of the workspace. The picture on the left shows the workspace of the system with two areas of interest and the regions generated by the discretisation. In each discrete region s_i, local policies are computed, one for each adjacent region, to drive the system from the region s_i to the contiguous regions s_i^1, s_i^2 and s_i^3.

The iMDP algorithm [23] approximates the continuous dynamics of the system using a sequence of Markov decision processes (MDPs) $\mathscr{M}_n = (Z_n, U, P_n, G_n, H_n)$ for $n \geq 0$, where Z_n is a discrete subset of X, U is the original control space, $P_n(\cdot|\cdot,\cdot) : Z_n \times Z_n \times U \to [0,1]$ gives the probability of transitioning to the state $x_j \in Z_n$ from the state $x_i \in Z_n$ under action $u \in U$, $G_n : Z_n \times U \to \mathbb{R}$ is an immediate cost function and $H_n : Z_n \to \mathbb{R}$ is a terminal cost function.

In each iteration of the algorithm, a new $\mathscr{M}_n$ is created by adding randomly sampled states to the set Z_{n-1} from the interior and boundary of the region s_i. To each state $x \in Z_n$, a non-negative interpolation interval $\Delta t_n(x)$, a cost value $J_n(x)$ and a control $u \in U$ are assigned. The interval $\Delta t_n(x)$, also called holding time, is used to approximate the discrete MDP $\mathscr{M}_n$ to the continuous system. Let $\{\chi_i^n, i \in \mathbb{Z}_+\}$ be a controlled Markov chain on $\mathscr{M}_n$ with probability transition P_n and let $\Delta\chi_i^n = \chi_{i+1} - \chi_i$ denote the distance between two consecutive states. In order to maintain the properties of the original system, $\Delta t_n(x)$ and P_n need to satisfy the following local consistency properties [18]:

- For all $x \in Z_n$:
$$\lim_{n\to\infty} \Delta t_n(x) = 0. \tag{4}$$
- For all $x_i \in Z_n$ and $u_i \in U$:
$$\mathbb{E}(\Delta\chi_i^n) = f(x_i, u_i)\Delta t_n(x_i) + o(\Delta t_n(x_i)), \tag{5}$$
$$\mathbb{E}([\Delta\chi_i^n - \mathbb{E}(\Delta\chi_i^n)][\Delta\chi_i^n - \mathbb{E}(\Delta\chi_i^n)]) = G(x_i)G(x_i)^T\Delta t_n(x_i) + o(\Delta t_n(x_i)), \tag{6}$$

$$\lim_{n\to\infty}\sup_{i\in\mathbb{Z}_+}\|\Delta\chi_i^n\| = 0, \tag{7}$$

where $\mathbb{E}$ is the conditional expectation given $\chi_i^n = x_i$ and $o(\cdot)$ indicates an upper bound on the error due to the discrete time approximation. The holding time $\Delta t_n(x)$ assigned to a state $x \in Z_n$ is computed as follows:

$$\Delta t_n(x) = \gamma\left(\frac{\log|Z_n|}{|Z_n|}\right)^{\theta\varsigma\rho/d_x}, \tag{8}$$

where $\gamma > 0$, $\theta \in (0,1]$, $\varsigma \in (0,1)$ and $\rho \in (0,1]$ are constants [23].

Recall that each region $s_i \in S$ requires one policy for each adjacent region. In order to compute a policy to drive the system from the interior of s_i to a particular contiguous region, say s_i^1, a negative terminal cost is assigned to states sampled from the boundary shared with s_i^1. To avoid the non-desired adjacent regions, s_i^2 and s_i^3, a positive terminal cost is assigned to states sampled from the boundary shared with these regions. A policy is defined by a function μ_n that maps each state $x \in Z_n$ to a control $u \in U$. Let $\mathbf{U}$ be the set of all possible polices. The optimal policy is found by minimising the cost-to-go function [23]:

$$J_{n,\mu_n}(x) = \mathbb{E}_{P_n}\left[\sum_{i=0}^{T_n-1}\alpha^{t_i^n}G_n(\chi_i^n,\mu_n(\chi_i^n)) + \alpha^{t_{T_n}^n}H_n(\chi_{T_n}^n)\right], \tag{9}$$

where $t_i^n = \sum_0^{i-1}\Delta t_n(\chi_i^n)$, $\alpha \in [0,1)$ is the discount rate, $\mathbb{E}_{P_n}$ is the conditional expectation given $\chi_0^n = x$ under P_n and T_n is the expected first exit of the controlled Markov chain $\{\chi_i^n, i \in \mathbb{Z}_+\}$ under the policy $\mu_n \in \mathbf{U}$ from the region s_i. The optimal policy μ_n^* satisfies $J_{n,\mu_n^*}(x) = \inf_{\mu_n\in\mathbf{U}} J_{n,\mu_n}(x)$.

The policy μ_n^* is used to assign a control value $\mu_n^*(x)$ to each non-boundary state $x \in Z_n$. This process is repeated to obtain a local optimal policy for each adjacent region. Because of the Delaunay triangulation, the discrete regions are triangles. Therefore, each region s_i has three local policies, denoted as $\mu_{s_i}^1$, $\mu_{s_i}^2$ and $\mu_{s_i}^3$, to drive the system from the interior of s_i to the adjacent regions s_i^1, s_i^2 and s_i^3. Different partitioning would lead to a different number of local policies.

B. BMDP Model

The probability of ending in an adjacent region s_j under a policy $\{\mu_{s_i}^l, l \in \{1,2,3\}\}$ varies among the sampled states x within the region s_i. Hence, the probability of transitioning from the region s_i to the region s_j is given by a range. To model this range, an BMDP $\mathscr{B} = (S, A, \hat{P}, \check{P}, L_S)$ is utilised. The set of states S is the set of regions created by the Delaunay triangulation and A is a set of actions. For clarity, we refer to states $s \in S$ as regions. The available actions in each region $s \in S$ are denoted by $A(s)$. Since each region has three local policies, actions $a_{s_i}^1$, $a_{s_i}^2$, $a_{s_i}^3 \in A(s_i)$ correspond to the local policies $\mu_{s_i}^1$, $\mu_{s_i}^2$, and $\mu_{s_i}^3$, respectively. $\hat{P}$ and $\check{P}$, defined as in Section II.B, are calculated as follows:

$$\hat{P}(s_j|s_i,a_{s_i}^l) = \max_{x\in Z_n} P(s_j|x,\mu_{s_i}^l), \tag{10}$$

$$\check{P}(s_j|s_i,a_{s_i}^l) = \min_{x\in Z_n} P(s_j|x,\mu_{s_i}^l), \tag{11}$$

where $P(s_j|x,\mu_{s_i}^l)$ is the probability of state x inside region s_i to finish in the region s_j when the local policy $\mu_{s_i}^l$ is applied. Since the Markov chain $\{\chi_i, i \in \mathbb{Z}_+\}$, induced by the policy $\mu_{s_i}^l$, is absorbing [24], these probabilities can be computed using the fundamental matrix [25]. The label function $L_S : S \to \Pi$ maps each state x within a region s to the set of atomic propositions Π. In order to select an action, or local policy, in each region such that the probability of satisfying a specification φ is maximised, a Cartesian product between the BMDP, described above, and a timed automaton that represents φ is created. This process is explained in the next subsection.

C. Product BMDP

This subsection explains the construction of the product BMDP $\mathscr{P}$ between the BMDP $\mathscr{B}$ and the timed automaton $\mathscr{T}_\varphi$ that represents the MITL formula φ. In order to obtain a discrete time space for the BMDP abstraction $\mathscr{B}$, we discretise the range of the clocks in C as follows. For each clock $c_i \in C$, let c_i^r be the maximum value in the range of clock c_i and let $\Delta\tau_i = \frac{c_i^r}{W_i}$, where $W_i \in \mathbb{R}_+$ is a constant such that $c_i^r \equiv 0\ (mod\ W_i)$. Then, the range of each clock $c_i \in C$ is divided into time intervals of the form: $[\kappa_i\Delta\tau_i, (\kappa_i+1)\Delta\tau_i - \varepsilon]$, where $0 \le \kappa_i \le \frac{c_i^r}{W_i}$ and ε is a small positive number. Similar to the clock vector $\mathbf{C}$, we define, with abuse of notation, the vector $\mathbf{T}$ with entries equal to the interval containing the value of each clock, i.e., $\mathbf{T}[i] = [\tau_i^a, \tau_i^b]$ such that $\tau_i^a \le \mathbf{C}[i] \le \tau_i^b$ for $i \in \{1,\dots,d_c\}$. The set of all possible vectors $\mathbf{T}$ is denoted by $\mathbb{T}$. Finally, we introduce the modified timed automaton $\mathscr{T}^\tau = (\Sigma, Q^\tau, q_0^\tau, Q_F^\tau, C, \Lambda(C), \to)$, where:

- $\Sigma = 2^\Pi$ is a finite alphabet,
- Q^τ is a finite set of states,
- $q_0^\tau \in Q^\tau$ is an initial state,
- $Q_F^\tau \subset Q^\tau$ is a set of accepting states,
- C is a finite set of clocks,
- $\Lambda(C)$ is a set of clock constraints,
- $\to \subseteq Q^\tau \times Q^\tau \times \Sigma \times 2^C \times \Lambda(C)$ is a transition relation.

Configurations in $\mathscr{T}^\tau$ are defined by pairs $(q, \mathbf{T})$. A run in $\mathscr{T}^\tau$ on a timed word (σ, η) is an infinite sequence $(q_0, \mathbf{T}_0) \xrightarrow{\sigma_1,\Delta\eta_1} (q_1, \mathbf{T}_1) \xrightarrow{\sigma_2,\Delta\eta_2} \dots$, where $\mathbf{T}_0[i] = [0, \Delta\tau_i - \varepsilon]$ for $i \in \{1,\dots,d_c\}$ and for all $i \ge 1$, $\mathbf{C}_i[j] = \mathbf{C}_{i-1}[j] + \Delta\eta_i$ if $c_j \notin \delta$, $\mathbf{C}_i[j] = 0$ if $c_j \in \delta$ and $\mathbf{C}[i] \in \mathbf{T}[i]$. Similar to a run in the timed automaton $\mathscr{T}$, a timed word (σ, η) is accepted by $\mathscr{T}^\tau$ if a state $q \in Q_F^\tau$ is visited in the run produced by (σ, η).

The construction of the product BMDP $\mathscr{P}$ is now presented. Given the timed automaton $\mathscr{T}_\varphi^\tau$ representing the formula φ and the BMDP $\mathscr{B}$, the product BMDP $\mathscr{P} = \mathscr{B} \times \mathscr{T}_\varphi^\tau$ is defined by the tuple $\mathscr{P} = (S_\mathscr{P}, A_\mathscr{P}, L_\mathscr{P}, \hat{P}_\mathscr{P}, \check{P}_\mathscr{P}, F_\mathscr{P})$, where:

- $S_\mathscr{P} = S \times Q^\tau \times \mathbb{T}$ is a finite set of states,
- $A_\mathscr{P} = A$,

- $L_{\mathscr{P}} = L_S$,
- $\hat{P}_{\mathscr{P}}((s',q',\mathbf{T}')|(s,q,\mathbf{T}),a_s^l)) = \hat{P}(s'|s,a_s^l)$ iff $(q,\mathbf{T}) \xrightarrow{L_{\mathscr{P}}(s'),\Delta\tau} (q',\mathbf{T}')$ and 0 otherwise,
- $\check{P}_{\mathscr{P}}((s',q',\mathbf{T}')|(s,q,\mathbf{T}),a_s^l)) = \check{P}(s'|s,a_s^l)$ iff $(q,\mathbf{T}) \xrightarrow{L_{\mathscr{P}}(s'),\Delta\tau} (q',\mathbf{T}')$ and 0 otherwise,
- $F_{\mathscr{P}} = S \times Q_F^{\tau} \times \mathbb{T}$ is a set of accepting states.

Each transition $(q,\mathbf{T}) \xrightarrow{L_{\mathscr{P}}(s'),\Delta\tau} (q',\mathbf{T}')$, where $\Delta\tau$ represents an increment of $\Delta\tau_i$ in the endpoints of the interval of each clock $c_i \in C$, satisfies the clock constraints as follows. A clock constraint of the form $c_i \leq k$ is satisfied by $\mathbf{T}'$ if $\tau_i^b \leq k$, where $\mathbf{T}'[i] = [\tau_i^a, \tau_i^b]$. On the other hand, a clock constraint $c_i \geq k$ is satisfied by $\mathbf{T}'$ if $\tau_i^a \geq k$.

Once the product BMDP $\mathscr{P}$ is created, a solution to the problem described in Section II.E can be computed as shown in the next subsection.

D. Optimal Global Policy Computation

In [12], the authors prove that the probability of satisfying the specification φ in the discrete semantics is equal to the probability of the controlled Markov chain $\{\chi_i, i \in \mathbb{Z}_+\}$ on $\mathscr{P}$, induced by a policy $\mu_{\mathscr{P}}$, reaching the set of final states $F_{\mathscr{P}}$. In this subsection, the computation of the policy $\mu_{\mathscr{P}} : S_{\mathscr{P}} \to A_{\mathscr{P}}$ is presented.

To find the policy that maximises the probability of reaching $F_{\mathscr{P}}$, the Interval Value Iteration (IVI) algorithm [19] is utilised. This algorithm can optimise a value function using the lower bound $\check{P}_{\mathscr{P}}$ or the upper bound $\hat{P}_{\mathscr{P}}$. In [19], these are referred to as pessimistic and optimistic value functions, respectively. In this paper, the pessimistic value function is utilised. The algorithm maximises the value function:

$$V(s_{\mathscr{P},i}) = \max_{a_s^l \in A_{\mathscr{P}}(s_{\mathscr{P},i})} \min_{\bar{P} \in [\check{P}_{\mathscr{P}}, \hat{P}_{\mathscr{P}}]} \sum_{s_{\mathscr{P},j} \in S_{\mathscr{P}}} \bar{P}(s_{\mathscr{P},j}|s_{\mathscr{P},i}, a_s^l) V(s_{\mathscr{P},j}), \tag{12}$$

for all $s_{\mathscr{P},i} \in \{S_{\mathscr{P}} \setminus F_{\mathscr{P}}\}$ and $V(s_{\mathscr{P},i}) = 1$ for all $s_{\mathscr{P},i} \in F_{\mathscr{P}}$. Intuitively, the value $V(s_{\mathscr{P},i})$ is the probability of reaching the set of final states $F_{\mathscr{P}}$ starting from $s_{\mathscr{P},i} \in S_{\mathscr{P}}$. Projected to the discrete approximation of the system, $V(s_{\mathscr{P},i})$ represents the worst-case probability of satisfying the specification φ from the states x within the region s_i given that $s_{\mathscr{P},i} = (s_i,\cdot,\cdot)$. Hence, the policy that maximises $V(s_{\mathscr{P},i})$, denoted by $\mu_{\mathscr{P}}^*$, is selected as an optimal global policy for the product BMDP $\mathscr{P}$.

E. Policy Implementation

The computed optimal global and local policies are implemented in the following manner. Given the initial system state $x(t)$ with $t = 0$ and the clock vector $\mathbf{C}$ with all entries equal to zero, the product BMDP state $s_{\mathscr{P}} = (s,q,\mathbf{T}) \in S_{\mathscr{P}}$ that satisfies: (i) $x(t) \in s^0$, i.e., $x(t)$ is in the interior of the region s, (ii) $q = q_0^{\tau}$ and (iii) $\mathbf{T}[i] = [0, \Delta\tau_i - \varepsilon]$ for all $i \in \{1,\ldots,d_c\}$, is identified. The local policy μ_s^l that corresponds to the optimal action $\mu_{\mathscr{P}}^*(s_{\mathscr{P}})$ is selected to control the system. To apply a local policy, the nearest sampled state $x_{nearest}$, in the interior of region s, to the current system state is sought. Then, the control $\mu_s^l(x_{nearest})$ is applied for $\Delta t_n(x_{nearest})$ units of time. At the next state $x(t')$, where $t' = t + \Delta t_n(x_{nearest})$, the clocks $c \notin \delta$ are incremented by $\Delta t_n(x_{nearest})$ units of time, i.e., $\mathbf{C}'[i] = \mathbf{C}[i] + \Delta t_n(x_{nearest})$ if $c_i \notin \delta$ and $\mathbf{C}'[i] = 0$ if $c_i \in \delta$. The new product BMDP state $(s',q',\mathbf{T}') \in S_{\mathscr{P}}$ satisfying: (i) $x(t') \in s'^0$, (ii) $q \xrightarrow{L_{\mathscr{P}}(s'),\Delta t_n(x_{nearest})} q'$ and (iii) $\mathbf{T}'[i] = [\tau_i^a, \tau_i^b]$ such that $\tau_i^a \leq \mathbf{C}'[i] \leq \tau_i^b$ for all $i \in \{1,\ldots,d_c\}$, is selected and the process is repeated.

IV. Analysis

A. Convergence

In this subsection, the convergence of the probability of satisfying a specification under the computed policy is analysed. In [12], the authors prove that as the discretisation in time and space becomes finer, the probability of satisfying a specification φ under the policy computed in the discrete approximation converges to the probability of the continuous system satisfying φ. This convergence is proved for a policy obtained in a product automaton between a DTA and a Markov chain approximating the continuous system. Two main points differentiate the Cartesian product used in [12] and the one utilised in this paper: (i) scalar values represent the value of each clock on the configurations of the DTA in contrast with the intervals in this paper and (ii) the value function of states in the product, i.e., $V(s_{\mathscr{P}})$, is a unique scalar value in [12], whereas in this paper, it is a value within the interval $[\check{V}(s_{\mathscr{P}}), \hat{V}(s_{\mathscr{P}})]$ given by the pessimistic and optimistic case, see Section III.D. Nevertheless, due to the following conditions, the proof of convergence in [12] can be applied to the method proposed in this paper.

First, it is shown in [17] that as the size of the regions shrink to zero, the error introduced in the computation of local policies converges to zero if the local consistency properties, described in Section III.A, are satisfied. Moreover, as the size of the regions decreases, the probability of satisfying φ from all the sampled states x within the region s converges to a single value [17]. Formally, the following condition holds:

$$\lim_{\text{diameter}(s)\to 0} \Big[\hat{V}(s_{\mathscr{P}}) - \check{V}(s_{\mathscr{P}}) = 0\Big]. \tag{13}$$

Now let $\Delta\tau_i \to 0$ for all $i \in \{1,\ldots,d_c\}$ in the timed automaton ${\mathscr{T}^{\tau}}_{\varphi}$. As the size of the intervals tends to zero, each configuration in ${\mathscr{T}^{\tau}}_{\varphi}$ is equivalent to an unique time instant in the trajectory of the system as in [12]. Hence, as the size of the regions and the size of the intervals approximate zero, the proof of Theorem 2 in [12] holds for the discrete approximation presented in this paper.

B. Complexity

The complexity of the proposed algorithm can be divided into two parts: the computation of local policies in each region, phase 1, and the computation of the global policy in the product BMDP, phase 2. The iMDP algorithm used to find local policies has a time complexity $O(|Z_n|^{1+\theta} \log|Z_n|)$, where Z_n is the set of sampled states within a region and $\theta \in (0,1]$ is a constant [23]. On the other hand, the number

of iterations of the IVI algorithm, required to converge to an optimal interval value, is polynomial in the number of states in the product BMDP $\mathscr{P}$ [19], which has at most $|S| \times |Q^\tau| \times |\mathbb{T}|$ states, where S, Q^τ and $\mathbb{T}$ are the set of regions, the number of states in the timed automaton $\mathscr{T}^\tau$ and the set of all possible vectors $\mathbf{T}$, respectively. The complexity of constructing a deterministic timed automaton from MTL formulae can be found in [26]. Note that the dynamics of the system are only considered in the first phase. Hence, for complex dynamics, the computation time on the the first phase would increase, but the time required to find a global policy is polynomial in the number of discrete regions.

V. EXAMPLE

The proposed approach is illustrated in the following example. We considered a two-dimensional system modelled as:

$$f(x(t),u(t)) = u, \quad G(x(t)) = 0.05I_2, \tag{14}$$

where $u \in [-0.5, 0.5]$ and I_2 is the identity matrix of 2×2. In this example the workspace is constrained by $0 \leq x \leq 3$ and $0 \leq y \leq 3$ and has two areas of interest marked by the atomic propositions π_1 and π_2, Figure 2. The objective is to maximise the probability of satisfying the MITL specification $\varphi = \Diamond_{[0,20]}(\pi_1 \wedge \Diamond_{[10,20]}(\pi_2))$, which indicates that the system has to reach areas π_1 and π_2 in maximum 20 units of time with the restriction of visiting π_2 after the tenth unit of time, similar to the example presented in [12].

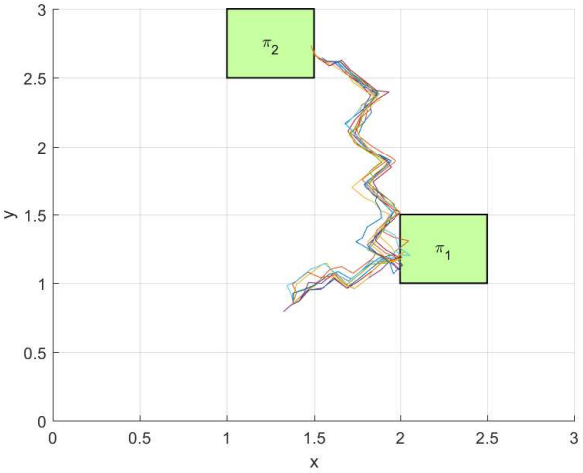

Fig. 2: Illustration of 10 sample paths of the system in (14). The system has to visit region π_1 and π_2 within 20 units of time, nevertheless, region π_2 has to be visited after the tenth unit of time. Formally the specification can be written as $\varphi = \Diamond_{[0,20]}(\pi_1 \wedge \Diamond_{[10,20]}(\pi_2))$.

For this example, the workspace is partitioned in 132 discrete regions by Triangle [27] and the size of the intervals, i.e., $\Delta\tau$, is 0.5. This discretisation produces a product BMDP $\mathscr{P}$ with 10516 states. To compute local policies, 300 discrete states are randomly sampled in each region. This process requires 3716 seconds. On the other hand, the construction of $\mathscr{P}$ and the computation of the optimal global policy require, on average, 4468 seconds. The probability of satisfying the specification under the computed policy is .8097. On average, the system reaches π_2 in 18.97 seconds, Figure 3. The example above is implemented in MATLAB on a desktop with a 3.30 GHz processor i5 and 8 GB of RAM.

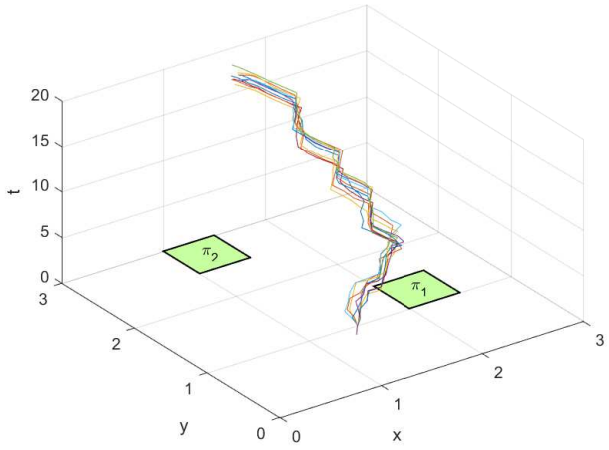

Fig. 3: 3D view of 10 trajectories of the system in (14) following the MITL specification $\varphi = \Diamond_{[0,20]}(\pi_1 \wedge \Diamond_{[10,20]}(\pi_2))$. The x, y and t axis show the position of the system and the time, respectively. The average time required to reach π_2 is 18.97 seconds.

A. Discussion

In this section we compare the proposed framework to the work in [12]. The first aspect to be compared is the required time to find a solution. The example above requires, on average, 8184 seconds to be solved. In contrast, for a similar specification, the method in [12] requires 19080 seconds. Although we considered a system with simpler dynamics, only the first phase of the method would be affected by a system with more complex dynamics. In Section IV.B it was shown that the complexity of computing local policies depends on the number of sampled states. Therefore, depending on the number of samples, the total required time to compute local and global policies could be larger compared to the time required in [12]. Nevertheless, for a reasonable number of samples, our method is faster as demonstrated in the example above. Moreover, for new MITL formulae, the method proposed in this paper would be always faster than [12]. This is achieved because only the second phase has to be solved. Formally, an optimal policy is obtained using a value iteration algorithm in the Cartesian product in both methods. Recall that the number of iterations of the algorithm, required to converge to an optimal value, is polynomial in the number of states. Since, in the proposed method, the dynamics of the system are reasoned in the first phase, the number of states in the Cartesian product depends only on the number of discrete regions of

the coarse segmentation. In contrast, in [12], the number of states depends on a finer discretisation of the state space.

The second aspect is the smoothness of the trajectory. It can be seen in Figure 2 that the trajectory shows a 'zigzag' pattern in contrast to the example in [12] where the trajectory is smoother. This pattern is caused by the local policies computed in each region. Since the local optimal policies are obtained by solving an optimisation problem, all the sampled states have assigned the control that produces the shortest internal path to the adjacent regions. Therefore, a quick change in the direction can be observed when the system reaches a new region. A possible solution is to reduce the size of the regions to obtain a finer segmentation. Nevertheless, this would have an impact in the time required to solve the problem. In other words, the method offers a trade-off between the smoothness of the trajectory and the time needed to find a solution.

VI. CONCLUSIONS

In this paper we have introduced a new method to find optimal policies based on metric interval temporal logic (MITL) for stochastic dynamic systems. Policies are optimal with respect to the probability of satisfying an MITL specification. In contrast to previous works, the motion of the continuous system is coarsely abstracted in a bounded-parameter Markov decision process. This allows a faster computation of policies. A main benefit of the method is that once local policies are computed, an optimal global policy can be found faster than current methods. The analysis shows that, as the discretisation gets finer, the probability of satisfying a specification under the computed policy converges to the probability of the continuous system satisfying the specification. A possible direction for future work includes the improvement of the smoothness of the trajectory without the necessity of a finer segmentation.

REFERENCES

[1] E. Aydin Gol, M. Lazar, and C. Belta, "Language-guided controller synthesis for discrete-time linear systems," in *Proc. of HSCC*. ACM, 2012, pp. 95–104.

[2] G. E. Fainekos, A. Girard, H. Kress-Gazit, and G. J. Pappas, "Temporal logic motion planning for dynamic robots," *Automatica*, vol. 45, no. 2, pp. 343–352, 2009.

[3] M. Kloetzer and C. Belta, "A fully automated framework for control of linear systems from temporal logic specifications," *Automatic Control, IEEE Transactions on*, vol. 53, no. 1, pp. 287–297, 2008.

[4] E. M. Wolff, U. Topcu, and R. M. Murray, "Optimal control of non-deterministic systems for a computationally efficient fragment of temporal logic," in *Proc. of CDC*. IEEE, 2013, pp. 3197–3204.

[5] X. C. Ding, S. L. Smith, C. Belta, and D. Rus, "MDP optimal control under temporal logic constraints," in *Proc. of CDC-ECC*. IEEE, 2011, pp. 532–538.

[6] B. Lacerda, D. Parker, and N. Hawes, "Optimal and dynamic planning for Markov decision processes with co-safe LTL specifications," in *Proc. of IROS*. IEEE, 2014, pp. 1511–1516.

[7] M. Svoreňová, J. Křetínský, M. Chmelík, K. Chatterjee, I. Černá, and C. Belta, "Temporal logic control for stochastic linear systems using abstraction refinement of probabilistic games," in *Proc. of HSCC*. ACM, 2015, pp. 259–268.

[8] M. B. Horowitz, E. M. Wolff, and R. M. Murray, "A compositional approach to stochastic optimal control with co-safe temporal logic specifications," in *Proc. of IROS*. IEEE, 2014, pp. 1466–1473.

[9] R. Koymans, "Specifying real-time properties with metric temporal logic," *Real-Time Systems*, vol. 2, no. 4, pp. 255–299, 1990.

[10] J. Liu and P. Prabhakar, "Switching control of dynamical systems from metric temporal logic specifications," in *Proc. of ICRA*. IEEE, 2014, pp. 5333–5338.

[11] S. S. Farahani, V. Raman, and R. M. Murray, "Robust model predictive control for signal temporal logic synthesis," *IFAC-PapersOnLine*, vol. 48, no. 27, pp. 323–328, 2015.

[12] J. Fu and U. Topcu, "Computational methods for stochastic control with metric interval temporal logic specifications," in *Proc of CDC*. IEEE, 2015.

[13] S. Karaman and E. Frazzoli, "Sampling-based motion planning with deterministic μ-calculus specifications," in *Proc. of CDC/CCC*. IEEE, 2009, pp. 2222–2229.

[14] A. Bhatia, L. E. Kavraki, and M. Y. Vardi, "Sampling-based motion planning with temporal goals," in *Proc. of ICRA*. IEEE, 2010, pp. 2689–2696.

[15] C. I. Vasile and C. Belta, "Sampling-based temporal logic path planning," in *Proc. of IROS*. IEEE, 2013, pp. 4817–4822.

[16] M. Lahijanian, L. E. Kavraki, and M. Y. Vardi, "A sampling-based strategy planner for nondeterministic hybrid systems," in *Proc. of ICRA*. IEEE, 2014, pp. 3005–3012.

[17] R. Luna, M. Lahijanian, M. Moll, and L. E. Kavraki, "Asymptotically optimal stochastic motion planning with temporal goals," in *Algorithmic Foundations of Robotics XI*. Springer, 2015, pp. 335–352.

[18] H. Kushner and P. G. Dupuis, *Numerical methods for stochastic control problems in continuous time*. Springer Science & Business Media, 2013, vol. 24.

[19] R. Givan, S. Leach, and T. Dean, "Bounded-parameter Markov decision processes," *Artificial Intelligence*, vol. 122, no. 1, pp. 71–109, 2000.

[20] R. Alur, T. Feder, and T. A. Henzinger, "The benefits of relaxing punctuality," *Journal of the ACM (JACM)*, vol. 43, no. 1, pp. 116–146, 1996.

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

[22] R. Alur and D. L. Dill, "A theory of timed automata," *Theoretical Computer Science*, vol. 126, no. 2, pp. 183–235, 1994.

[23] V. A. Huynh, S. Karaman, and E. Frazzoli, "An incremental sampling-based algorithm for stochastic optimal control," in *Proc. of ICRA*. IEEE, 2012, pp. 2865–2872.

[24] V. A. Huynh, "Sampling-based algorithms for stochastic optimal control," Ph.D. dissertation, Massachusetts Institute of Technology, 2014.

[25] J. G. Kemeny, J. L. Snell *et al.*, *Finite Markov chains*. van Nostrand Princeton, NJ, 1960, vol. 356.

[26] D. Ničković and N. Piterman, *From MTL to deterministic timed automata*. Springer, 2010.

[27] J. R. Shewchuk, "Triangle: Engineering a 2D quality mesh generator and delaunay triangulator," in *Applied Computational Geometry: Towards Geometric Engineering*. Springer-Verlag, 1996, vol. 1148, pp. 203–222.