52nd IEEE Conference on Decision and Control
December 10-13, 2013. Florence, Italy

Optimal Control of MDPs with Temporal Logic Constraints

Mária Svoreňová, Ivana Černá and Calin Belta

***Abstract*— In this paper, we focus on formal synthesis of control policies for finite Markov decision processes with non-negative real-valued costs. We develop an algorithm to automatically generate a policy that guarantees the satisfaction of a correctness specification expressed as a formula of Linear Temporal Logic, while at the same time minimizing the expected average cost between two consecutive satisfactions of a desired property. The existing solutions to this problem are sub-optimal. By leveraging ideas from automata-based model checking and game theory, we provide an optimal solution. We demonstrate the approach on an illustrative example.**

I. INTRODUCTION

Markov Decision Processes (MDP) are probabilistic models widely used in various areas, such as economics, biology, and engineering. In robotics, they have been successfully used to model the motion of systems with actuation and sensing uncertainty, such as ground robots [17], unmanned aircraft [22], and surgical steering needles [1]. MDPs are central to control theory [4], probabilistic model checking and synthesis in formal methods [3], [9], and game theory [13].

MDP control is a well studied area (see *e.g.,* [4]). The goal is usually to optimize the expected value of a cost over a finite time (*e.g.,* stochastic shortest path problem) or an average expected cost in infinite time (*e.g.,* average cost per stage problem). Recently, there has been increasing interest in developing MDP control strategies from rich specifications given as formulas of probabilistic temporal logics, such as Probabilistic Computation Tree Logic (PCTL) and Probabilistic Linear Temporal Logic (PLTL) [12], [17]. It is important to note that both optimal control and temporal logic control problems for MDPs have their counterpart in automata game theory. Specifically, optimal control translates to solving $1^1/_2$-player games with payoff functions, such as discounted-payoff and mean-payoff games [6]. Temporal logic control for MDPs corresponds to solving $1^1/_2$-player games with parity objectives [2].

Our aim is to optimize the behavior of a system subject to correctness (temporal logic) constraints. Such a connection between optimal and temporal logic control is an intriguing problem with potentially high impact in several applications. Consider, for example, a mobile robot involved in a persistent surveillance mission in a dangerous area under tight fuel or time constraints. The correctness requirement is expressed as a temporal logic specification, *e.g.,* "Keep visiting A and then B and always avoid C". The resource constraints translate to minimizing a cost function over the feasible trajectories of the robot. Motivated by such applications, in this paper we focus on correctness specifications given as LTL formulae and optimization objectives expressed as average expected cumulative costs per surveillance cycle (ACPC).

The main contribution of this work is to provide a sound and complete solution to the above problem. This paper can be seen as an extension of [18], [20], [11], [8]. In [18], we focused on deterministic transition systems and developed a finite-horizon online planner to provably satisfy an LTL constraint while optimizing the behavior of the system between every two consecutive satisfactions of a given proposition. We extended this framework in [20], where we optimize the long-term average behavior of deterministic transition systems with time-varying events of known statistics. The closest to this work is [11], where the authors focus on a problem of optimal LTL control of MDPs with real-valued costs on actions. The correctness specification is assumed to include a persistent surveillance task and the goal is to minimize the long-term expected average cost between successive visits of the locations under surveillance. Using dynamic programming techniques, the authors design a solution that is sub-optimal in the general case. In [8], it is shown that, for a certain fragment of LTL, the solution becomes optimal. By using recent results from game theory [5], in this paper we provide an optimal solution for full LTL. Due to space limitations, we only include the main results and omit all the proofs and complexity analysis. These can be found in the technical report [19].

II. PRELIMINARIES

For a set S, we use S^ω and S^+ to denote the set of all infinite and all non-empty finite sequences of elements of S, respectively. For a sequence $\tau = a_0 \dots a_n \in \mathsf{S}^+$, $|\tau| = n+1$ denotes the length of τ. For $0 \leq i \leq n$, $\tau(i) = a_i$ and $\tau^{(i)} = \mathsf{a_0} \dots \mathsf{a_i}$ is the prefix of τ of length $i+1$. We use the same notation for infinite sequences from S^ω.

Definition 1: A *Markov decision process* (MDP) is a tuple $\mathcal{M} = (S, A, \mathbf{P}, \mathrm{AP}, L, g)$, where S is a non-empty finite set of states, A is a non-empty finite set of actions, $\mathbf{P}\colon S \times A \times S \to [0,1]$ is a transition probability function such that for every state $s \in S$ and action $\alpha \in A$ it holds that $\sum_{s' \in S} \mathbf{P}(s,\alpha,s') \in \{0,1\}$, AP is a finite set of atomic propositions, $L\colon S \to 2^{\mathrm{AP}}$ is a labeling function, and $g\colon S \times A \to \mathbb{R}_0^+$ is a cost function. An *initialized* MDP is an MDP with a distinctive initial state $s_{init} \in S$.

An action $\alpha \in A$ is called enabled in a state $s \in S$ if $\sum_{s' \in S} \mathbf{P}(s,\alpha,s') = 1$. With a slight abuse of notation, $A(s)$ denotes the set of all actions enabled in a state s. We assume $A(s) \neq \emptyset$ for every $s \in S$.

M. Svoreňová, I. Černá are with Faculty of Informatics, Masaryk University, Brno, Czech Republic, svorenova@mail.muni.cz, cerna@muni.cz. C. Belta is with Department of Mechanical Engineering and the Division of Systems Engineering, Boston University, Boston, MA, USA, cbelta@bu.edu. This work was partially supported at Masaryk University by grants GAP202/11/0312, LH11065, and at Boston University by ONR grants MURI N00014-09-1051, MURI N00014-10-10952 and by NSF grant CNS-1035588.

A run of an MDP $\mathcal{M}$ is an infinite sequence of states $\rho = s_0 s_1 \ldots \in S^\omega$ such that for every $i \geq 0$, there exists $\alpha_i \in A(s_i)$, $\mathbf{P}(s_i, \alpha_i, s_{i+1}) > 0$. We use $\mathrm{Run}^{\mathcal{M}}(s)$ to denote the set of all runs of $\mathcal{M}$ that start in a state $s \in S$. Let $\mathrm{Run}^{\mathcal{M}} = \bigcup_{s \in S} \mathrm{Run}^{\mathcal{M}}(s)$. A finite run $\sigma = s_0 \ldots s_n \in S^+$ of $\mathcal{M}$ is a finite prefix of a run in $\mathcal{M}$ and $\mathrm{Run}^{\mathcal{M}}_{\mathrm{fin}}(s)$ denotes the set of all finite runs of $\mathcal{M}$ starting in a state $s \in S$. Let $\mathrm{Run}^{\mathcal{M}}_{\mathrm{fin}} = \bigcup_{s \in S} \mathrm{Run}^{\mathcal{M}}_{\mathrm{fin}}(s)$. The length $|\sigma| = n+1$ of a finite run $\sigma = s_0 \ldots s_n$ is also referred to as the number of stages of the run. The last state of σ is denoted by $last(\sigma) = s_n$.

The word induced by a run $\rho = s_0 s_1 \ldots$ of $\mathcal{M}$ is an infinite sequence $L(s_0)L(s_1)\ldots \in (2^{\mathrm{AP}})^\omega$. Similarly, a finite run of $\mathcal{M}$ induces a finite word from the set $(2^{\mathrm{AP}})^+$.

Definition 2: Let $\mathcal{M} = (S, A, \mathbf{P}, \mathrm{AP}, L, g)$ be an MDP. An *end component* (EC) of the MDP $\mathcal{M}$ is an MDP $\mathcal{N} = (S_{\mathcal{N}}, A_{\mathcal{N}}, \mathbf{P}|_{\mathcal{N}}, \mathrm{AP}, L|_{\mathcal{N}}, g|_{\mathcal{N}})$ such that $\emptyset \neq S_{\mathcal{N}} \subseteq S$, $\emptyset \neq A_{\mathcal{N}} \subseteq A$. For every $s \in S_{\mathcal{N}}$ and $\alpha \in A_{\mathcal{N}}(s)$ it holds that $\{s' \in S \mid \mathbf{P}(s, \alpha, s') > 0\} \subseteq S_{\mathcal{N}}$. For every pair of states $s, s' \in S_{\mathcal{N}}$, there exists a finite run $\sigma \in \mathrm{Run}^{\mathcal{N}}_{\mathrm{fin}}(s)$ such that $last(\sigma) = s'$. We use $\mathbf{P}|_{\mathcal{N}}$ to denote the function $\mathbf{P}$ restricted to the sets $S_{\mathcal{N}}$ and $A_{\mathcal{N}}$. Similarly, we use $L|_{\mathcal{N}}$ and $g|_{\mathcal{N}}$ with the obvious meaning. If the context is clear, we only use $\mathbf{P}, L, g$ instead of $\mathbf{P}|_{\mathcal{N}}, L|_{\mathcal{N}}, g|_{\mathcal{N}}$. EC $\mathcal{N}$ of $\mathcal{M}$ is called *maximal* (MEC) if there is no EC $\mathcal{N}' = (S_{\mathcal{N}'}, A_{\mathcal{N}'}, \mathbf{P}, \mathrm{AP}, L, g)$ of $\mathcal{M}$ such that $\mathcal{N}' \neq \mathcal{N}$, $S_{\mathcal{N}} \subseteq S_{\mathcal{N}'}$ and $A_{\mathcal{N}}(s) \subseteq A_{\mathcal{N}'}(s)$ for every $s \in S_{\mathcal{N}}$. The set of all end components and maximal end components of $\mathcal{M}$ are denoted by $\mathrm{EC}(\mathcal{M})$ and $\mathrm{MEC}(\mathcal{M})$, respectively.

The number of ECs of an MDP $\mathcal{M}$ can be up to exponential in the number of states of $\mathcal{M}$ and they can intersect. On the other hand, MECs are pairwise disjoint and every EC is contained in a single MEC. Hence, the number of MECs of $\mathcal{M}$ is bounded by the number of states of $\mathcal{M}$.

Definition 3: Let $\mathcal{M} = (S, A, \mathbf{P}, \mathrm{AP}, L, g)$ be an MDP. A *control strategy* for $\mathcal{M}$ is a function $C \colon \mathrm{Run}^{\mathcal{M}}_{\mathrm{fin}} \to A$ such that for every $\sigma \in \mathrm{Run}^{\mathcal{M}}_{\mathrm{fin}}$ it holds that $C(\sigma) \in A(last(\sigma))$.

A strategy C for which $C(\sigma) = C(\sigma')$ for all finite runs $\sigma, \sigma' \in \mathrm{Run}^{\mathcal{M}}_{\mathrm{fin}}$ with $last(\sigma) = last(\sigma')$ is called memoryless. In that case, we consider C to be a function $C \colon S \to A$. A strategy is called finite-memory if it is defined as a tuple $C = (M, \mathsf{act}, \Delta, \mathsf{start})$, where M is a finite set of modes, $\Delta \colon M \times S \to M$ is a transition function, $\mathsf{act} \colon M \times S \to A$ selects an action to be applied in $\mathcal{M}$, and $\mathsf{start} \colon S \to M$ selects the starting mode for every $s \in S$.

A run $\rho = s_0 s_1 \ldots \in \mathrm{Run}^{\mathcal{M}}$ of an MDP $\mathcal{M}$ is called a run under a strategy C for $\mathcal{M}$ if for every $i \geq 0$, it holds that $\mathbf{P}(s_i, C(\rho^{(i)}), s_{i+1}) > 0$. A finite run under C is a finite prefix of a run under C. The set of all infinite and finite runs of $\mathcal{M}$ under C starting in a state $s \in S$ are denoted by $\mathrm{Run}^{\mathcal{M},C}(s)$ and $\mathrm{Run}^{\mathcal{M},C}_{\mathrm{fin}}(s)$, respectively. Let $\mathrm{Run}^{\mathcal{M},C} = \bigcup_{s \in S} \mathrm{Run}^{\mathcal{M},C}(s)$ and $\mathrm{Run}^{\mathcal{M},C}_{\mathrm{fin}} = \bigcup_{s \in S} \mathrm{Run}^{\mathcal{M},C}_{\mathrm{fin}}(s)$.

Let $\mathcal{M}$ be an MDP, s a state of $\mathcal{M}$, and C a strategy for $\mathcal{M}$. There exists a unique probability measure $\Pr^{\mathcal{M},C}_s$ such that, given a subset $X \subseteq \mathrm{Run}^{\mathcal{M},C}(s)$, $\Pr^{\mathcal{M},C}_s(X)$ is the probability that a run of $\mathcal{M}$ under C that starts in s belongs to the set X.

The following properties hold for any MDP $\mathcal{M}$ (see, *e.g.,* [3]). For every EC $\mathcal{N}$ of $\mathcal{M}$, there exists a finite-memory strategy C for $\mathcal{M}$ such that $\mathcal{M}$ under C starting from any state of $\mathcal{N}$ never visits a state outside $\mathcal{N}$ and all states of $\mathcal{N}$ are visited infinitely many times with probability 1. On the other hand, having any, finite-memory or not, strategy C, a state s of $\mathcal{M}$ and a run ρ of $\mathcal{M}$ under C that starts in s, the set of states visited infinitely many times by ρ forms an end component. Let $\mathsf{ec} \subseteq \mathrm{EC}(\mathcal{M})$ be the set of all ECs of $\mathcal{M}$ that correspond, in the above sense, to at least one run under the strategy C that starts in the state s. We say that the strategy C leads $\mathcal{M}$ from the state s to the set ec.

Definition 4: Linear Temporal Logic (LTL) formulae over a set AP are formed according to the following grammar:

$$\phi ::= true \mid a \mid \neg\phi \mid \phi \wedge \phi \mid \mathbf{X}\,\phi \mid \phi\,\mathbf{U}\,\phi \mid \mathbf{G}\,\phi \mid \mathbf{F}\,\phi,$$

where $a \in \mathrm{AP}$, $\neg$, $\wedge$ are standard Boolean connectives, and $\mathbf{X}$ (*next*), $\mathbf{U}$ (*until*), $\mathbf{G}$ (*always*), and $\mathbf{F}$ (*eventually*) are temporal operators.

Formulae of LTL are interpreted over the words from $(2^{\mathrm{AP}})^\omega$, such as those induced by runs of an MDP $\mathcal{M}$ (for details see *e.g.,* [3]). For example, a word $w \in (2^{\mathrm{AP}})^\omega$ satisfies $\mathbf{G}\,\phi$ and $\mathbf{F}\,\phi$ if ϕ holds in w always and eventually, respectively. If the word induced by a run $\rho \in \mathrm{Run}^{\mathcal{M}}$ satisfies a formula ϕ, we say that the run ρ satisfies ϕ. With slight abuse of notation, we also use states or sets of states of the MDP as propositions in LTL formulae.

For every LTL formula ϕ, the set of all runs of $\mathcal{M}$ that satisfy ϕ is measurable in the probability measure $\Pr^{\mathcal{M},C}_s$ for any C and s [3]. With slight abuse of notation, we use LTL formulae as arguments of $\Pr^{\mathcal{M},C}_s$. If for a state $s \in S$ it holds that $\Pr^{\mathcal{M},C}_s(\phi) = 1$, we say that the strategy C almost-surely satisfies ϕ starting from s. If $\mathcal{M}$ is an initialized MDP and $\Pr^{\mathcal{M},C}_{s_{init}}(\phi) = 1$, we say that C almost-surely satisfies ϕ.

The LTL control synthesis problem for an initialized MDP $\mathcal{M}$ and an LTL formula ϕ over AP aims to find a strategy for $\mathcal{M}$ that almost-surely satisfies ϕ. This problem can be solved using principles from probabilistic model checking [3], [12]. The algorithm itself is based on the translation of ϕ to a Rabin automaton and the analysis of an MDP that combines the Rabin automaton and the original MDP $\mathcal{M}$.

Definition 5: A *deterministic Rabin automaton* (DRA) is a tuple $\mathcal{A} = (Q, 2^{\mathrm{AP}}, \delta, q_0, Acc)$, where Q is a non-empty finite set of states, 2^{AP} is an alphabet, $\delta \colon Q \times 2^{\mathrm{AP}} \to Q$ is a transition function, $q_0 \in Q$ is an initial state, and $Acc \subseteq 2^Q \times 2^Q$ is an accepting condition.

A run of $\mathcal{A}$ is a sequence $q_0 q_1 \ldots \in Q^\omega$ such that for every $i \geq 0$, there exists $A_i \in 2^{\mathrm{AP}}$, $\delta(q_i, A_i) = q_{i+1}$. We say that the word $A_0 A_1 \ldots \in (2^{\mathrm{AP}})^\omega$ induces the run $q_0 q_1 \ldots$. A run of $\mathcal{A}$ is called accepting if there exists a pair $(B, G) \in Acc$ such that the run visits every state from B only finitely many times and at least one state from G infinitely many times.

For every LTL formula ϕ over AP, there exists a DRA $\mathcal{A}_\phi$ such that all and only words from $(2^{AP})^\omega$ satisfying ϕ induce an accepting run of $\mathcal{A}_\phi$ [14]. For translation algorithms see *e.g.,* [16], and their online implementations, *e.g.,* [15].

Definition 6: Let $\mathcal{M} = (S, A, \mathbf{P}, \mathrm{AP}, L, g)$ be an initialized MDP and $\mathcal{A} = (Q, 2^{\mathrm{AP}}, \delta, q_0, Acc)$ be a DRA. The *product* of $\mathcal{M}$ and $\mathcal{A}$ is the initialized MDP $\mathcal{P} = (S_{\mathcal{P}}, A, \mathbf{P}_{\mathcal{P}}, \mathrm{AP}_{\mathcal{P}}, L_{\mathcal{P}}, g_{\mathcal{P}})$, where $S_{\mathcal{P}} = S \times Q$,

$\mathbf{P}_{\mathcal{P}}((s,q),\alpha,(s',q')) = \mathbf{P}(s,\alpha,s')$ if $q' = \delta(q,L(s))$ and 0 otherwise, $\mathrm{AP}_{\mathcal{P}} = Q$, $L_{\mathcal{P}}((s,q)) = q$, $g_{\mathcal{P}}((s,q),\alpha) = g(s,\alpha)$. The initial state of $\mathcal{P}$ is $s_{\mathcal{P}init} = (s_{init}, q_0)$.

Using the projection on the first component, every (finite) run of $\mathcal{P}$ projects to a (finite) run of $\mathcal{M}$ and vice versa, for every (finite) run of $\mathcal{M}$, there exists a (finite) run of $\mathcal{P}$ that projects to it. Analogous correspondence exists between strategies for $\mathcal{P}$ and $\mathcal{M}$. It holds that the projection of a finite-memory strategy for $\mathcal{P}$ is also finite-memory. More importantly, for the product $\mathcal{P}$ of an MDP $\mathcal{M}$ and a DRA $\mathcal{A}_\phi$ for an LTL formula ϕ, the probability of satisfying the accepting condition Acc of $\mathcal{A}_\phi$ under a strategy $C_{\mathcal{P}}$ for $\mathcal{P}$ starting from the initial state $s_{\mathcal{P}init}$ is equal to the probability of satisfying the formula ϕ in the MDP $\mathcal{M}$ under the projected strategy C starting from the initial state s_{init}.

Definition 7: Let $\mathcal{P} = (S_{\mathcal{P}}, A, \mathbf{P}_{\mathcal{P}}, \mathrm{AP}_{\mathcal{P}}, L_{\mathcal{P}}, g_{\mathcal{P}})$ be the product of an MDP $\mathcal{M}$ and a DRA $\mathcal{A}$. An *accepting end component* (AEC) of $\mathcal{P}$ is defined as an end component $\mathcal{N} = (S_{\mathcal{N}}, A_{\mathcal{N}}, \mathbf{P}_{\mathcal{P}}, \mathrm{AP}_{\mathcal{P}}, L_{\mathcal{P}}, g_{\mathcal{P}})$ of $\mathcal{P}$ for which there exists a pair (B,G) in the acceptance condition of $\mathcal{A}$ such that $L_{\mathcal{P}}(S_{\mathcal{N}}) \cap B = \emptyset$ and $L_{\mathcal{P}}(S_{\mathcal{N}}) \cap G \neq \emptyset$. We say that $\mathcal{N}$ is accepting with respect to the pair (B,G). An AEC $\mathcal{N} = (S_{\mathcal{N}}, A_{\mathcal{N}}, \mathbf{P}_{\mathcal{P}}, \mathrm{AP}_{\mathcal{P}}, L_{\mathcal{P}}, g_{\mathcal{P}})$ is called *maximal* (MAEC) if there is no AEC $\mathcal{N}' = (S_{\mathcal{N}'}, A_{\mathcal{N}'}, \mathbf{P}_{\mathcal{P}}, \mathrm{AP}_{\mathcal{P}}, L_{\mathcal{P}}, g_{\mathcal{P}})$ such that $\mathcal{N}' \neq \mathcal{N}$, $S_{\mathcal{N}} \subseteq S_{\mathcal{N}'}$, $A_{\mathcal{N}}((s,q)) \subseteq A_{\mathcal{N}'}((s,q))$ for every $(s,q) \in S_{\mathcal{P}}$ and $\mathcal{N}$ and $\mathcal{N}'$ are accepting with respect to the same pair. We use $\mathrm{AEC}(\mathcal{P})$ and $\mathrm{MAEC}(\mathcal{P})$ to denote the set of all accepting end components and maximal accepting end components of $\mathcal{P}$, respectively.

Note that MAECs that are accepting with respect to the same pair are always disjoint. However, MAECs that are accepting with respect to different pairs can intersect.

III. Problem Formulation

Consider an initialized MDP $\mathcal{M} = (S, A, \mathbf{P}, \mathrm{AP}, L, g)$ and an LTL formula ϕ over AP of the form

$$\phi = \varphi \wedge \mathbf{GF}\,\pi_{\mathrm{sur}}, \tag{1}$$

where $\pi_{\mathrm{sur}} \in \mathrm{AP}$ is an atomic proposition and φ is an LTL formula over AP. Intuitively, a formula of such form states two partial goals – mission goal φ and surveillance goal $\mathbf{GF}\,\pi_{\mathrm{sur}}$. To satisfy the whole formula the system must accomplish the mission and visit the surveillance states $S_{\mathrm{sur}} = \{s \in S \mid \pi_{\mathrm{sur}} \in L(s)\}$ infinitely many times. The motivation for this form of specification comes from applications in robotics, where persistent surveillance tasks are often a part of the specification. Note that the form in Eq. (1) does not restrict the full LTL expressivity since every LTL formula ϕ_1 can be translated into a formula ϕ_2 of the form in Eq. (1) that is associated with the same set of runs of $\mathcal{M}$. Explicitly, $\phi_2 = \phi_1 \wedge \mathbf{GF}\,\pi_{\mathrm{sur}}$, where π_{sur} is such that $\pi_{\mathrm{sur}} \in L(s)$ for every state $s \in S$.

In this work, we focus on a control synthesis problem, where the goal is to almost-surely satisfy a given LTL specification, while optimizing a long-term quantitative objective. The objective is to minimize the average expected cumulative cost between consecutive visits to surveillance states.

Formally, we say that every visit to a surveillance state completes a surveillance cycle. In particular, starting from the initial state, the first visit to S_{sur} completes the first surveillance cycle of a run. We use $\sharp(\sigma)$ to denote the number of completed surveillance cycles in a finite run σ plus one. For a strategy C for $\mathcal{M}$, the cumulative cost in the first n stages of applying C to $\mathcal{M}$ starting from a state $s \in S$ is

$$g_{\mathcal{M},C}(s,n) = \sum_{i=0}^{n} g(\sigma_{s,n}^{\mathcal{M},C}(i), C(\sigma_{s,n}^{\mathcal{M},C^{(i)}})),$$

where $\sigma_{s,n}^{\mathcal{M},C}$ is the random variable whose values are finite runs of length $n+1$ from the set $\mathrm{Run}_{\mathrm{fin}}^{\mathcal{M},C}(s)$ and the probability of a finite run σ is $\Pr_s^{\mathcal{M},C}(\mathrm{Cyl}(\sigma))$. Note that $g_{\mathcal{M},C}(s,n)$ is also a random variable. Finally, we define the average expected cumulative cost per surveillance cycle (ACPC) in the MDP $\mathcal{M}$ under a strategy C as a function $V_{\mathcal{M},C} \colon S \to \mathbb{R}_0^+$ such that for a state $s \in S$

$$V_{\mathcal{M},C}(s) = \limsup_{n\to\infty} E\Big(\frac{g_{\mathcal{M},C}(s,n)}{\sharp(\sigma_{s,n}^{\mathcal{M},C})}\Big).$$

The problem we consider in this paper is then the following.

Problem 1: Let $\mathcal{M} = (S, A, \mathbf{P}, \mathrm{AP}, L, g)$ be an initialized MDP, ϕ be an LTL formula over AP of the form in Eq. (1). Find a strategy C for $\mathcal{M}$ such that C almost-surely satisfies ϕ and, at the same time, C minimizes the ACPC value $V_{\mathcal{M},C}(s_{init})$ among all strategies almost-surely satisfying ϕ.

The above problem was recently investigated in [11]. However, the solution presented by the authors is guaranteed to find an optimal strategy only if every MAEC $\mathcal{N}$ of the product $\mathcal{P}$ of the MDP $\mathcal{M}$ and the DRA for the specification satisfies certain conditions (for details see [11]). In this paper, we present a solution to Problem 1 that always finds an optimal strategy if one exists. The algorithm is based on principles from probabilistic model checking [3] and game theory [5], whereas the authors in [11] mainly use results from dynamic programming [4].

In the special case when every state of $\mathcal{M}$ is a surveillance state, Problem 1 aims to find a strategy that minimizes the average expected cost per stage among all strategies almost-surely satisfying ϕ. The problem of minimizing the average expected cost per stage (ACPS) in an MDP, without considering any correctness specification, is a well studied problem in optimal control, see *e.g.,* [4]. It holds that there always exists a stationary strategy that minimizes the ACPS value starting from the initial state. In our approach to Problem 1, we use techniques for solving the ACPS problem to find a strategy that minimizes the ACPC value.

IV. Solution

Let $\mathcal{M} = (S, A, \mathbf{P}, \mathrm{AP}, L, g)$ be an initialized MDP and ϕ an LTL formula over AP of the form in Eq. (1). To solve Problem 1 for $\mathcal{M}$ and ϕ we leverage ideas from game theory [5] and construct an optimal strategy for $\mathcal{M}$ as a combination of a strategy that ensures the almost-sure satisfaction of the specification ϕ and a strategy that guarantees the minimum ACPC value among all strategies that do not cause immediate unrepairable violation of ϕ.

The algorithm we present in this section works with the product $\mathcal{P} = (S_{\mathcal{P}}, A, \mathbf{P}_{\mathcal{P}}, \mathrm{AP}_{\mathcal{P}}, L_{\mathcal{P}}, g_{\mathcal{P}})$ of the MDP $\mathcal{M}$ and a deterministic Rabin automaton $\mathcal{A}_\phi = (Q, 2^{\mathrm{AP}}, \delta, q_0, Acc)$

for the formula ϕ. We inherit the notion of a surveillance cycle in $\mathcal{P}$ by adding the proposition π_{sur} to the set $\text{AP}_{\mathcal{P}}$ and to the set $L_{\mathcal{P}}((s,q))$ for every $(s,q) \in S_{\mathcal{P}}$ such that $\pi_{\text{sur}} \in L(s)$. Using the correspondence between strategies for $\mathcal{P}$ and $\mathcal{M}$, an optimal strategy C for $\mathcal{M}$ is found as a projection of a strategy $C_{\mathcal{P}}$ for $\mathcal{P}$ which almost-surely satisfies the accepting condition Acc of $\mathcal{A}_\phi$ and at the same time, minimizes the ACPC value $V_{\mathcal{P},C_{\mathcal{P}}}(s_{\mathcal{P}init})$ among all strategies for $\mathcal{P}$ that almost-surely satisfy Acc.

Since $C_{\mathcal{P}}$ must almost-surely satisfy the accepting condition Acc, it leads from the initial state of $\mathcal{P}$ to a set of MAECs. For every MAEC $\mathcal{N}$, the minimum ACPC value $V^*_{\mathcal{N}}((s,q))$ that can be obtained in $\mathcal{N}$ starting from a state $(s,q) \in S_{\mathcal{N}}$ is equal for all the states of $\mathcal{N}$ and we denote this value $V^*_{\mathcal{N}}$. The strategy $C_{\mathcal{P}}$ is constructed in two steps.

First, we find a set maec^* of MAECs of $\mathcal{P}$ and a strategy C_0 that leads $\mathcal{P}$ from the initial state to the set maec^*. We require that C_0 and maec^* minimize the weighted average of the values $V^*_{\mathcal{N}}$ for $\mathcal{N} \in \mathsf{maec}^*$. The strategy $C_{\mathcal{P}}$ applies C_0 from the initial state until $\mathcal{P}$ enters the set maec^*.

Second, we solve the problem of how to control the product once a state of an MAEC $\mathcal{N} \in \mathsf{maec}^*$ is visited. Intuitively, we combine two finite-memory strategies, $C^\phi_{\mathcal{N}}$ for the almost-sure satisfaction of the accepting condition Acc and $C^V_{\mathcal{N}}$ for maintaining the average expected cumulative cost per surveillance cycle. To satisfy both objectives, the strategy $C_{\mathcal{P}}$ is played in rounds. In each round, we first apply the strategy $C^\phi_{\mathcal{N}}$ and then the strategy $C^V_{\mathcal{N}}$, each for a specific (finite) number of steps.

A. Finding an optimal set of MAECs

Let $\text{MAEC}(\mathcal{P})$ be the set of all MAECs of the product $\mathcal{P}$ that can be computed as follows. For every pair $(B,G) \in Acc$, we create a new MDP from $\mathcal{P}$ by removing all its states with label in B and the corresponding actions. For the new MDP, we use one of the algorithms in [10], [9], [7] to compute the set of all its MECs. Finally, for every MEC, we check whether it contains a state with label in G.

In this section, the aim is to find a set $\mathsf{maec}^* \subseteq \text{MAEC}(\mathcal{P})$ and a strategy C_0 for $\mathcal{P}$ that satisfy conditions formally stated below. Since the strategy C_0 will only be used to enter the set maec^*, it is constructed as a partial function.

Definition 8: A *partial strategy* ζ for the MDP $\mathcal{M}$ is a partial function $\zeta\colon \text{Run}^{\mathcal{P}}_{\text{fin}} \to A$, where if $\zeta(\sigma)$ is defined for $\sigma \in \text{Run}^{\mathcal{P}}_{\text{fin}}$, then $\zeta(\sigma) \in A(last(\sigma))$.

A partial stationary strategy for $\mathcal{M}$ can also be considered as a partial function $\zeta\colon S \to A$ or a subset $\zeta \subseteq S \times A$. The set $\text{Run}^{\mathcal{M},\zeta}$ of runs of $\mathcal{M}$ under ζ contains all infinite runs of $\mathcal{M}$ that follow ζ and all those finite runs σ of $\mathcal{M}$ under ζ for which $\zeta(last(\sigma))$ is not defined. A finite run of $\mathcal{M}$ under ζ is then a finite prefix of a run under ζ. The probability measure $\Pr^{\mathcal{M},\zeta}_s$ is defined in the same manner as in Sec. II. We also extend the semantics of LTL formulas to finite words. For example, a formula $\mathbf{FG}\,\phi$ is satisfied by a finite word if in some non-empty suffix of the word ϕ always holds.

The conditions on maec^* and C_0 are as follows. The partial strategy C_0 leads $\mathcal{P}$ to the set maec^*, and we require that maec^* and C_0 minimize the value

$$\sum_{\mathcal{N} \in \mathsf{maec}^*} \Pr^{\mathcal{P},C_0}_{s_{\mathcal{P}init}}(\mathbf{FG}\, S_{\mathcal{N}}) \cdot V^*_{\mathcal{N}}.$$

The procedure to compute the optimal ACPC value $V^*_{\mathcal{N}}$ for an MAEC $\mathcal{N}$ of $\mathcal{P}$ is presented in the next section. Assume we already computed this value for each MAEC. The algorithm to find the set maec^* and partial strategy C_0 is a straightforward reduction to one of the basic optimization problems for MDPs, the stochastic shortest path problem, see *e.g.,* [4].

B. Optimizing ACPC value in an MAEC

In this section, we compute the minimum ACPC value $V^*_{\mathcal{N}}$ that can be attained in an MAEC $\mathcal{N} \in \text{MAEC}(\mathcal{P})$ and construct the corresponding strategy for $\mathcal{N}$.

Essentially, we reduce the problem of computing the minimum ACPC value to the problem of computing the minimum ACPS value by reducing $\mathcal{N}$ to an MDP such that every state of the reduced MDP is labeled with the surveillance proposition π_{sur}.

Let $\mathcal{N} = (S_{\mathcal{N}}, A_{\mathcal{N}}, \mathbf{P}_{\mathcal{P}}, \text{AP}_{\mathcal{P}}, L_{\mathcal{P}}, g_{\mathcal{P}})$ be an MAEC of $\mathcal{P}$. Since it is an MAEC, there exists a state $(s,q) \in S_{\mathcal{N}}$ with $\pi_{\text{sur}} \in L_{\mathcal{P}}((s,q))$. Let $S_{\mathcal{N}_{\text{sur}}}$ denote the set of all such states in $S_{\mathcal{N}}$. We reduce $\mathcal{N}$ to an MDP

$$\mathcal{N}_{\text{sur}} = (S_{\mathcal{N}_{\text{sur}}}, \mathbf{A}_{\text{sur}}, \mathbf{P}_{\text{sur}}, \text{AP}_{\mathcal{P}}, L_{\mathcal{P}}, g_{\text{sur}})$$

using Alg. 1. For the sake of readability, we use singletons such as v instead of pairs such as (s,q) to denote the states of $\mathcal{N}$. The MDP $\mathcal{N}_{\text{sur}}$ is constructed from $\mathcal{N}$ by eliminating states from $S_{\mathcal{N}} \backslash S_{\mathcal{N}_{\text{sur}}}$ one by one in arbitrary order. The actions $\mathbf{A}_{\text{sur}}$ are partial stationary strategies for $\mathcal{N}$ in which we remember all the states and actions we eliminated. Later we prove that the transition probability $\mathbf{P}_{\text{sur}}(v,\zeta,v')$ for states $v, v' \in S_{\mathcal{N}_{\text{sur}}}$ and an action $\zeta \in \mathbf{A}_{\text{sur}}(v)$ is the probability that in $\mathcal{N}$ under the partial stationary strategy ζ, if we start from the state v, the next state that will be visited from the set $S_{\mathcal{N}_{\text{sur}}}$ is the state v', *i.e.,* the first surveillance cycle is completed by visiting v'. The cost $g_{\text{sur}}(v,\zeta)$ is the expected cumulative cost gained in $\mathcal{N}$ using partial stationary strategy ζ from v until we reach a state in $S_{\mathcal{N}_{\text{sur}}}$.

Proposition 1: Let $\mathcal{N} = (S_{\mathcal{N}}, A_{\mathcal{N}}, \mathbf{P}_{\mathcal{P}}, \text{AP}_{\mathcal{P}}, L_{\mathcal{P}}, g_{\mathcal{P}})$ be an MAEC and $\mathcal{N}_{\text{sur}} = (S_{\mathcal{N}_{\text{sur}}}, \mathbf{A}_{\text{sur}}, \mathbf{P}_{\text{sur}}, \text{AP}_{\mathcal{P}}, L_{\mathcal{P}}, g_{\text{sur}})$ its reduction resulting from Alg. 1. The minimum ACPC value that can be attained in $\mathcal{N}_{\text{sur}}$ starting from any of its states is the same and we denote it $V^*_{\mathcal{N}_{\text{sur}}}$. There exists a stationary strategy $C^V_{\mathcal{N}_{\text{sur}}}$ for $\mathcal{N}_{\text{sur}}$ that attains this value regardless of the starting state in $\mathcal{N}_{\text{sur}}$. Both $V^*_{\mathcal{N}_{\text{sur}}}$ and $C^V_{\mathcal{N}_{\text{sur}}}$ can be computed as a solution to the ACPS problem for $\mathcal{N}_{\text{sur}}$. It holds that $V^*_{\mathcal{N}} = V^*_{\mathcal{N}_{\text{sur}}}$ and from $C^V_{\mathcal{N}_{\text{sur}}}$, one can construct a finite-memory strategy $C^V_{\mathcal{N}}$ for $\mathcal{N}$ which regardless of the starting state in $\mathcal{N}$ attains the optimal ACPC value $V^*_{\mathcal{N}}$.

The following property of the strategy $C^V_{\mathcal{N}}$ is crucial for the correctness of our approach to Problem 1.

Proposition 2: For every $(s,q) \in S_{\mathcal{N}}$, it holds that

$$\lim_{n\to\infty} \Pr^{\mathcal{N},C^V_{\mathcal{N}}}_{(s,q)}(\{\rho \mid \frac{g_{\mathcal{P}}(\rho^{(\sharp n)})}{n} \leq V^*_{\mathcal{N}}\}) = 1,$$

where $g_{\mathcal{P}}(\rho^{(\sharp n)})$ denotes the cumulative cost gained in the first n surveillance cycles of a run $\rho \in \text{Run}^{\mathcal{N}}((s,q))$. Hence,

Algorithm 1 Reduction of an MAEC $\mathcal{N}$ to $\mathcal{N}_{\text{sur}}$

Input: $\mathcal{N} = (S_{\mathcal{N}}, A_{\mathcal{N}}, \mathbf{P}_{\mathcal{P}}, \text{AP}_{\mathcal{P}}, L_{\mathcal{P}}, g_{\mathcal{P}})$
Output: $\mathcal{N}_{\text{sur}} = (S_{\mathcal{N}_{\text{sur}}}, \mathbf{A}_{\text{sur}}, \mathbf{P}_{\text{sur}}, \text{AP}_{\mathcal{P}}, L_{\mathcal{P}}, g_{\text{sur}})$

let $X = (S_X, \mathbf{A}_X, \mathbf{P}_X, \text{AP}_{\mathcal{P}}, L_{\mathcal{P}}, g_X)$ be an MDP where
- $S_X := S_{\mathcal{N}}$,
- for $v \in S_X$: $\mathbf{A}_X(v) := \{\zeta_\alpha \mid \zeta_\alpha = \{(v, \alpha)\}, \alpha \in A_{\mathcal{N}}(v)\}$,
- for $v, v' \in S_X, \zeta \in \mathbf{A}_X$: $\mathbf{P}_X(v, \zeta, v') := \mathbf{P}_{\mathcal{P}}(v, \zeta(v), v')$,
- for $v \in S_X, \zeta \in \mathbf{A}_X$: $g_X(v, \zeta) := g_{\mathcal{P}}(v, \zeta(v))$

while $S_X \backslash S_{\mathcal{N}_{\text{sur}}} \neq \emptyset$ **do**
  let $v \in S_X \backslash S_{\mathcal{N}_{\text{sur}}}$
  for all $\zeta \in \mathbf{A}_X(v)$ **do**
    if $\mathbf{P}_X(v, \zeta, v) < 1$ **then**
      for all $v_{from} \in S_X, \zeta_{old} \in \mathbf{A}_X(v_{from})$ **do**
        if $\mathbf{P}_X(v_{from}, \zeta_{old}, v) > 0$ and ζ_{old}, ζ do not conflict for any state from S_X **then**
          $\zeta_{new} := \zeta_{old} \cup \zeta$
          add ζ_{new} to $\mathbf{A}_X(v_{from})$
          for every $v_{to} \in S_X$:

$$\mathbf{P}_X(v_{from}, \zeta_{new}, v_{to}) := \mathbf{P}_X(v_{from}, \zeta_{old}, v_{to}) + \\ + \mathbf{P}_X(v_{from}, \zeta_{old}, v) \cdot \frac{\mathbf{P}_X(v, \zeta, v_{to})}{1 - \mathbf{P}_X(v, \zeta, v)}$$

$$g_X(v_{from}, \zeta_{new}) := g_X(v_{from}, \zeta_{old}) + \\ + \mathbf{P}_X(v_{from}, \zeta_{old}, v) \cdot \frac{g_X(v, \zeta)}{1 - \mathbf{P}_X(v, \zeta, v)}$$

          remove ζ_{old} from $\mathbf{A}_X(v_{from})$
        end if
      end for
    end if
    remove ζ from $\mathbf{A}_X(v)$
  end for
  remove v from S_X
end while
return X

for every $\epsilon > 0$, there exists $j(\epsilon) \in \mathbb{N}$ such that if the strategy $C^V_{\mathcal{N}}$ is applied from a state $(s, q) \in S_{\mathcal{N}}$ for any $l \geq j(\epsilon)$ surveillance cycles, then the average expected cumulative cost per surveillance cycle in these l surveillance cycles is at most $V^*_{\mathcal{N}} + \epsilon$ with probability at least $1 - \epsilon$, *i.e.,*

$$\Pr^{\mathcal{N}, C^V_{\mathcal{N}}}_{(s,q)}(\{\rho \mid \frac{g_{\mathcal{P}}(\rho^{(\sharp l)})}{l} \leq V^*_{\mathcal{N}} + \epsilon\}) \geq 1 - \epsilon.$$

C. Almost-sure acceptance in an MAEC

Here we design a strategy for an $\mathcal{N} \in \text{MAEC}(\mathcal{P})$ that guarantees almost-sure satisfaction of the acceptance condition Acc of $\mathcal{A}_\phi$. Let (B, G) be a pair in Acc such that $\mathcal{N}$ is accepting with respect to (B, G), *i.e.,* $L_{\mathcal{P}}(S_{\mathcal{N}}) \cap B = \emptyset$ and $L_{\mathcal{P}}(S_{\mathcal{N}}) \cap G \neq \emptyset$. There exists a stationary strategy $C^\phi_{\mathcal{N}}$ for $\mathcal{N}$ under which a state with label in G is reached with probability 1 regardless of the starting state, *i.e.,*

$$\Pr^{\mathcal{N}, C^\phi_{\mathcal{N}}}_{(s,q)}(\mathbf{F}\, G) = 1$$

for every $(s, q) \in S_{\mathcal{N}}$. The existence of such a strategy follows from the fact that $\mathcal{N}$ is an EC [3]. Moreover, we construct $C^\phi_{\mathcal{N}}$ to minimize the expected cumulative cost before reaching a state in $S_{\mathcal{N}} \cap S \times G$. Similar to strategy C_0 from Sec. IV-A, strategy $C^\phi_{\mathcal{N}}$ can be computed using a reduction to the stochastic shortest path problem. For the full description of the reduction and proof of its correctness, we refer to the full version of this paper [19].

D. Optimal strategy for $\mathcal{P}$

Finally, we are ready to construct the strategy $C_{\mathcal{P}}$ for the product $\mathcal{P}$ that projects to an optimal solution for $\mathcal{M}$.

First, starting from the initial state $s_{\mathcal{P}init}$, $C_{\mathcal{P}}$ applies the strategy C_0 resulting from the algorithm described in Sec. IV-A until a state of an MAEC in the set maec^* is reached. Let $\mathcal{N} \in \mathsf{maec}^*$ denote the MAEC and let $(B, G) \in Acc$ be a pair from the accepting condition of $\mathcal{A}_\phi$ such that $\mathcal{N}$ is accepting with respect to (B, G).

Now, the strategy $C_{\mathcal{P}}$ starts to play the rounds. Each round consists of two phases. First, play the strategy $C^\phi_{\mathcal{N}}$ from Sec. IV-C until a state with label in G is reached. Let us denote k_i the number of steps we play $C^\phi_{\mathcal{N}}$ in i-th round. The second phase applies the strategy $C^V_{\mathcal{N}}$ from Sec. IV-B until the number of completed surveillance cycles in the second phase of the current round is l_i. The number l_i is any natural number for which

$$l_i \geq \max\{j(\tfrac{1}{i}), i \cdot k_i \cdot g_{\mathcal{P}max}\},$$

where $j(\frac{1}{i})$ is from Prop. 2 and $g_{\mathcal{P}max}$ is the maximum value of the costs $g_{\mathcal{P}}$. After applying the strategy $C^V_{\mathcal{N}}$ for l_i surveillance cycles, we proceed to the next round $i + 1$.

Theorem 1: The strategy $C_{\mathcal{P}}$ almost-surely satisfies the accepting condition Acc of $\mathcal{A}_\phi$ and at the same time, $C_{\mathcal{P}}$ minimizes the ACPC value $V_{\mathcal{P}, C_{\mathcal{P}}}(s_{\mathcal{P}init})$ among all strategies for $\mathcal{P}$ almost-surely satisfying Acc.

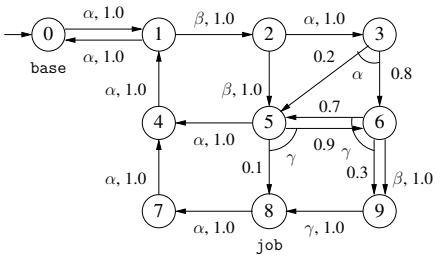
It is important to note that the resulting strategy $C_{\mathcal{P}}$ for $\mathcal{P}$, and hence the projected strategy C for $\mathcal{M}$ are not finite-memory strategies in general. The reason is that in the second phase of every round i, the strategy $C^V_{\mathcal{N}}$ is applied for l_i surveillance cycles and l_i is generally growing with i. This, however, does not prevent the solution to be effectively used. We introduce a simple rule to avoid performing all $l_i \geq \max\{i \cdot k_i \cdot g_{\mathcal{P}max}, j(\frac{1}{i})\}$ surveillance cycles in every round i. Intuitively, once the average cumulative cost per surveillance cycle in the second phase of a round is below a certain threshold, we can proceed in the computation to the next round. As the simulation results in Sec. V show, the use of this simple rule dramatically decreases the number of performed surveillance cycles in almost every round.

V. Case Study

We implemented the framework from Sec. IV in Java and applied it to a persistent surveillance robotics example [21]. In this section, we report on the simulation results.

Consider a mobile robot moving in a partitioned environment. The motion of the robot is modeled by the initialized MDP $\mathcal{M}$ shown in Fig. 1a. The set AP of atomic propositions contains two propositions `base` and `job`. As depicted in Fig. 1a, state 0 is the base location and state 8 is the job location. At the job location, the robot performs some work, and at the base, it reports on its job activity.

The robot's mission is to visit both base and job location infinitely many times. In addition, at least one job must be performed between every two visits of the base:
$\phi = \mathbf{GF}\,\texttt{base} \wedge \mathbf{GF}\,\texttt{job} \wedge \mathbf{G}\big(\texttt{base} \Rightarrow \mathbf{X}(\neg\texttt{base}\,\mathbf{U}\,\texttt{job})\big)$.
While satisfying the formula, we want to minimize the expected average cost between two consecutive jobs, *i.e.,* the surveillance proposition $\pi_{\text{sur}} = \texttt{job}$.



(a)

	Condition	0	1	2	3	4	5	6	7	8	9
C_{init}		α	–	–	–	–	–	–	–	–	–
C_{p1}	before `job`	α	β	α	α	α	γ	γ	α	α	γ
	after `job`	α	α	α	α	α	γ	γ	α	α	γ
C_{p2}		α	β	α	α	α	γ	γ	α	α	γ

(b)

Fig. 1: (a) Initialized MDP $\mathcal{M}$ with initial state 0. The costs of applying α, β, γ in any states are 5, 10, 1, respectively, *e.g.,* $g(1, \alpha) = 5$. (b) Definitions of strategies C_{init}, C_{p1}, C_{p2} for $\mathcal{M}$, the projections of strategies $C_0, C^{\phi}_{\mathcal{N}}, C^{V}_{\mathcal{N}}$ for $\mathcal{P}$, respectively. The condition "before `job`" means that the corresponding prescription is used if the job location has not yet been visited since the last visit of the base. Similarly, the prescription with condition "after `job`" is used if the job location was visited at least once since the last visit of the base.

The Rabin automaton $\mathcal{A}_\phi$ used in the simulation has 5 states and 1 pair in the accepting condition. The product $\mathcal{P}$ of $\mathcal{M}$ and $\mathcal{A}_\phi$ has 50 states and one MAEC $\mathcal{N}$ of 19 states. The optimal set of MAECs $\mathsf{maec}^* = \{\mathcal{N}\}$. The optimal ACPC value $V^*_{\mathcal{N}} = 40.5$. In Fig. 1b, we list the projections of strategies $C_0, C^{\phi}_{\mathcal{N}}, C^{V}_{\mathcal{N}}$ for $\mathcal{P}$ to strategies C_{init}, C_{p1}, C_{p2} for $\mathcal{M}$, respectively. The optimal strategy C for $\mathcal{M}$ is then defined as follows. Starting from the initial state 0, apply strategy C_{init} until a state is reached, where C_{init} is no longer defined. Start round number 1. In i-th round, proceed as follows. In the first phase, apply strategy C_{p1} (for k_i steps) until the base is reached and then for one more step (the product $\mathcal{P}$ has to reach a state from the Rabin pair). In the second phase, use strategy C_{p2} for $l_i = \max\{i \cdot k_i \cdot 10, j(\frac{1}{i})\}$ surveillance cycles, *i.e.,* until the number of jobs performed by the robot is l_i. We also apply the rule described in Sec. IV-D to shorten the second phase, if possible.

Let us summarize the statistical results we obtained for 5 executions of the strategy C for $\mathcal{M}$, each of 100 rounds. The number k_i of steps in the first phase of a round $i > 1$ was always 5 because in such case, the first phase starts at the job location and the strategy C_{p1} needs to be applied for exactly 4 steps to reach the base. Therefore, in every round $i > 1$, the number l_i is at least $50 \cdot i$, *e.g.,* in round 100, $l_i \geq 5000$. However, using the rule described in Sec. IV-D, the average number of jobs per round was 130 and the median was only 14. In particular, the number was not increasing with the round. On the contrary, it appears to be independent from the history of the execution. In addition, at most 2 rounds in each of the executions finished only at the point, when the number of jobs performed by the robot in the second phase reached l_i. The average ACPC value attained after 100 rounds was 40.56.

In contrast to our solution, the algorithm from [11] does not find an optimal strategy for $\mathcal{M}$. Regardless of the initialization, it always results in a sub-optimal strategy, namely the strategy C_{p1} from Fig. 1b that has ACPC value 50.5.

References

[1] R. Alterovitz, T. Siméon, and K. Goldberg. The stochastic motion roadmap: A sampling framework for planning with Markov motion uncertainty. In *Robotics: Science and Systems*. Citeseer, 2007.

[2] K. Apt and E. Grädel. *Lectures in Game Theory for Computer Scientists*. Cambridge University Press, 2011.

[3] C. Baier and J. Katoen. *Principles of model checking*. The MIT Press, 2008.

[4] D. Bertsekas. *Dynamic Programming and Optimal Control, vol. II*. Athena Scientific Optimization and Computation Series. Athena Scientific, 2007.

[5] K. Chatterjee and L. Doyen. Energy and Mean-Payoff Parity Markov Decision Processes. In *MFCS*, volume 6907 of *LNCS*, pages 206–218. 2011.

[6] K. Chatterjee and L. Doyen. Games and Markov Decision Processes with Mean-Payoff Parity and Energy Parity Objectives. In *MEMICS*, volume 7119 of *LNCS*, pages 37–46. 2012.

[7] K. Chatterjee and M. Henzinger. Faster and Dynamic Algorithms for Maximal End-Component Decomposition and Related Graph Problems in Probabilistic Verification. In *Proc. SODA*, pages 1318–1336, 2011.

[8] Y. Chen, J. Tumova, and C. Belta. LTL robot motion control based on automata learning of environmental dynamics. In *Proc. ICRA*, pages 5177–5182, 2012.

[9] C. Courcoubetis and M. Yannakakis. The complexity of probabilistic verification. *J. ACM*, 42(4):857–907, 1995.

[10] L. de Alfaro. *Formal Verification of Probabilistic Systems*. PhD thesis, Stanford University, 1997. Technical report STAN-CS-TR-98-1601.

[11] X. C. Ding, S. Smith., C. Belta, and D. Rus. MDP Optimal Control under Temporal Logic Constraints. In *Proc. CDC-ECC*, pages 532 –538, 2011.

[12] X. C. Ding, S. L. Smith, C. Belta, and D. Rus. LTL Control in Uncertain Environments with Probabilistic Satisfaction Guarantees. In *Proc. IFAC*, volume 18, 2011.

[13] J. Filar and K. Vrieze. *Competitive Markov Decision Processes*. Springer, 1996.

[14] E. Grädel, W. Thomas, and T. Wilke. *Automata, Logics, and Infinite Games: A Guide to Current Research*, volume 2500 of *LNCS*. Springer, 2002.

[15] J. Klein. ltl2dstar – LTL to Deterministic Streett and Rabin Automata, 2007. `http://www.ltl2dstar.de/`.

[16] J. Klein and C. Baier. Experiments with deterministic ω-automata for formulas of linear temporal logic. *Theoretical Computer Science*, 363(2):182 – 195, 2006.

[17] M. Lahijanian, S. B. Andersson, and C. Belta. Temporal logic motion planning and control with probabilistic satisfaction guarantees. *IEEE Transaction on Robotics*, 28:396–409, 2011.

[18] M. Svoreňová, J. Tůmová, J. Barnat, and I. Černá. Attraction-Based Receding Horizon Path Planning with Temporal Logic Constraints. In *Proc. CDC*, pages 6749–6754, 2012.

[19] M. Svoreňová, I. Černá, and C. Belta. Optimal Control of MDPs with Temporal Logic Constraints. Technical report, 2013. Available online at http://arxiv.org/abs/1303.1942.

[20] M. Svoreňová, I. Černá, and C. Belta. Optimal Receding Horizon Control for Finite Deterministic Systems with Temporal Logic Constraints. In *Proc. ACC*, 2013. To appear.

[21] M. Svoreňová, I. Černá, and C. Belta. Simulation of Optimal Control of MDPs with Temporal Logic Constraints, 2013. `http://www.fi.muni.cz/~x175388/simulationCDC13`.

[22] S. Temizer, M. J. Kochenderfer, L. P. Kaelbling, T. Lozano-Perez, and J. K. Kuchar. Collision Avoidance for Unmanned Aircraft using Markov Decision Processes. In *AIAA Guidance, Navigation and Control Conference*, 2010.