Optimal Temporal Logic Control for Deterministic Transition Systems With Probabilistic Penalties

Mária Svoreňová, *Student Member, IEEE*, Ivana Černá, and Calin Belta, *Senior Member, IEEE*

***Abstract*—We consider an optimal control problem for a weighted deterministic transition system required to satisfy a constraint expressed as a linear temporal logic (LTL) formula over its labels. By assuming that the executions of the system incur time-varying penalties modeled as Markov chains, our goal is to minimize the expected average cumulative penalty incurred between consecutive satisfactions of a desired property. Using concepts from theoretical computer science, we provide two solutions to this problem. First, we derive a provably correct optimal strategy within the class of strategies that do not exploit values of penalties sensed in real time. Second, we show that by taking advantage of locally sensing the penalties, we can construct heuristic strategies leading to lower collected penalty. While still ensuring satisfaction of the LTL constraint, we cannot guarantee optimality in the latter case. We provide a user-friendly implementation of the proposed algorithms and analysis of two case studies.**

***Index Terms*—Linear temporal logic (LTL), optimal control.**

I. Introduction

TEMPORAL logics, such as computation tree logic (CTL) and linear temporal logic (LTL), have been customarily used to specify correctness of computer programs and digital circuits modeled as finite-state transition systems [1]. The problem of analyzing such a model against a temporal logic formula, known as formal analysis or model checking, has received a lot of attention during the past 30 years, and efficient algorithms and software tools are available [2], [3]. On the other hand, the formal synthesis problem, in which the goal is to design or control a system from a temporal logic specification, has not been studied extensively until recently. Latest results include the use of model checking techniques in control of deterministic systems [4], games for controlling nondeterministic systems [5], linear programming and value iteration for controlling Markov decision processes [1], [6]. Through the use of abstractions, such techniques have also been used for infinite systems [7]–[12]. Alternatively, one can construct a discrete model using, e.g., [13]–[16], to which the above synthesis methods can be applied.

In formal synthesis, the connection between optimal and temporal logic control is an intriguing problem with a potentially high impact in applications. By combining the two areas, the goal is to optimize the behavior of a system subject to correctness constraints. For example, consider a mobile robot involved in a surveillance mission in a dangerous area and under tight fuel and time constraints. The correctness requirement is a temporal logic specification, e.g., "alternately keep visiting A and B and always avoid C," while the resource constraints translate to minimizing a cost function over the feasible trajectories. While optimal control is a mature discipline and formal synthesis is fairly well understood, optimal formal synthesis is a largely open area.

In this paper, we focus on finite weighted deterministic transition systems with correctness constraints given as formulas of LTL. We assume that every state of the system is associated with a time-varying penalty. The penalties can be used to encode dynamic features of the system such as energy or time demands for the mobile robot that change according to traffic load. The concept of such dynamic values is well adopted, e.g., in reinforcement learning [17] with applications in robotics, games or economics. Here, we consider probabilistic penalties defined as Markov chains that are used to model environmental phenomena with known statistics such as the traffic load or the value of a stock. Motivated by persistent surveillance robotic missions, our goal is to minimize the expected average cumulative penalty incurred between consecutive satisfactions of a property associated with some states of the system, while at the same time satisfying an additional temporal constraint. Also from robotics comes our assumption that in executions of the system, the penalty values can only be sensed locally in close proximity from the current state. We design two algorithms that bring together concepts from automata-based model checking, graph theory, and game theory. The first computes an offline, optimal control strategy that uses only the *a priori* known transition probabilities of the penalties' Markov chains, but does not exploit their actual values in real time. Up to special cases, the optimal strategy requires infinite memory. We show that using simple feedback, the strategy can be implemented efficiently. The second proposed algorithm shows that by taking advantage of the local sensing, we can design an online control strategy that, while still satisfying the temporal specification, provides lower value of the optimization function. The online strategy is a heuristic that locally improves the offline strategy

Manuscript received December 19, 2013; revised June 17, 2014 and September 14, 2014; accepted November 24, 2014. Date of publication December 18, 2014; date of current version May 21, 2015. This work was supported in part by Masaryk University under Grants GAP202/11/0312 and LH11065, in part by Boston University by the Office of Naval Research under Grants MURI N00014-09-1051 and MURI N00014-10-10952, and in part by the National Science Foundation under Grant CNS-1035588. Recommended by Associate Editor S. Takai.

M. Svoreňová and I. Černá are with the Faculty of Informatics, Masaryk University, 60200 Brno, Czech Republic (e-mail: svorenova@mail.muni.cz; cerna@muni.cz).

C. Belta is with the Department of Mechanical Engineering and Division of Systems Engineering, Boston University, Boston, MA 02215 USA (e-mail: cbelta@bu.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TAC.2014.2381451

TABLE I
List of the Most Frequently Used Symbols and Their Meaning

symbol	**meaning**
A, A^ω, A^+	set, set of all infinite and all nonempty finite sequences of elements of A
$\mathcal{T}, S, s_{init}$	transition system, its states and initial state
$\mathcal{B}, Q, q_0$	Büchi automaton, its states and initial state
$\mathcal{P}, S_\mathcal{P}, s_{\mathcal{P}init}$	product, its states and initial state
$C, C_\mathcal{P}$	strategy, strategy for product $\mathcal{P}$
$\sigma, \lvert\sigma\rvert, \sigma(i), \sigma^{(i)}, \sigma_C$	finite run, its length, i-th element and prefix of length $i+1$, $i \geq 0$; finite run under strategy C
$\rho, \rho(i), \rho^{(i)}, \rho_C$	infinite run, its i-th element and prefix of length $i+1$, $i \geq 0$; infinite run under strategy C
$g(s,t), g_E(s)$	penalty in state s at time t, expected value of penalty in state s
$g_{\text{sim}}(s,t,x,k)$	simulated expected value of penalty in state s at time $t+k$ given that its value at time t is x
$V_{\mathcal{T},C}(s)$	expected average cumulative penalty per surveillance cycle (APPC) for strategy C for $\mathcal{T}$ starting from s

based on local sensing and simulation over a user-defined planning horizon. While we can prove optimality of the offline strategy among the strategies that disregard local sensing, it is intractable to construct or use an optimal strategy among those that utilize it. We also suggest a method to construct a whole class of online strategies with good expected behaviors.

This paper is related to [4], [18]–[20], which also focus on optimal control for weighted deterministic transition systems with temporal constraints. In [4], the authors develop a control strategy that minimizes the maximum weight between consecutive visits to a given set of states, subject to constraints expressed as LTL formulas. In addition to weights, transitions in [18] are assigned with costs and a strategy is constructed that minimizes the weighted average cost, while satisfying an LTL formula. The proposed solution is related to our offline approach, where we disregard the local sensing of penalties and consider only their expected, static value. We address this correlation and our contribution over [18] in more detail in the following sections. Finally, time-varying, locally sensed rewards with unknown dynamics in the states of the system were introduced in [19]. The authors present an online receding horizon control strategy satisfying an LTL specification that is designed to maximize rewards collected locally. However, the solution does not have any global optimality guarantees. This approach was generalized in [20], where the authors assume an LTL specification that includes persistent surveillance. The objective is to maximize collected rewards between satisfactions of the surveillance property. As in [19], there are no optimality guarantees. Our contribution over [19], [20] is threefold. First, in this work we consider expected average behavior in infinite time instead of local finite horizon behavior. Second, we show that optimal or effective strategies can be constructed. Finally, we present two types of control strategies, offline and online, that offer different guarantees and results.

Preliminary results of this work appeared in [21], where we considered a particular penalty model that is only a special case of the Markov chain considered here. In addition to [21], this paper includes a detailed analysis between the offline and online strategies, a class of heuristic online strategies, an improved implementation, and several simulation results.

This paper is organized as follows. In Section II, we provide the necessary definitions. For readers' convenience, Table I lists the most frequently used symbols. The problem is formally stated in Section III and Section IV contains our main results. Finally, in Section V, we discuss simulation results.

II. Preliminaries

A. Deterministic Transition System

Definition 1: A weighted deterministic transition system (TS) is a tuple $\mathcal{T} = (S, T, AP, L, w)$, where S is a nonempty finite set of states, $T \subseteq S \times S$ is a transition relation, AP is a nonempty finite set of atomic propositions, $L : S \to 2^{AP}$ is a labeling function and $w : T \to \mathbb{R}^+$ is a weight function. We assume that for every $s \in S$ there exists $s' \in S$ such that $(s, s') \in T$. An initialized transition system is a TS $\mathcal{T} = (S, T, AP, L, w)$ with a distinctive initial state $s_{\text{init}} \in S$.

A run of a TS $\mathcal{T}$ is an infinite sequence $\rho = s_0 s_1 \ldots \in S^\omega$ such that for every $i \geq 0$ it holds $(s_i, s_{i+1}) \in T$. We use $\inf(\rho)$ to denote the set of all states visited infinitely many times in the run ρ and $\text{Run}^\mathcal{T}(s)$ for the set of all runs of $\mathcal{T}$ that start in $s \in S$. Let $\text{Run}^\mathcal{T} = \bigcup_{s \in S} \text{Run}^\mathcal{T}(s)$. A finite run $\sigma = s_0 \ldots s_n$ of $\mathcal{T}$ is a finite prefix of a run of $\mathcal{T}$ and $\text{Run}^\mathcal{T}_{\text{fin}}(s)$ denotes the set of all finite runs of $\mathcal{T}$ that start in $s \in S$. Let $\text{Run}^\mathcal{T}_{\text{fin}} = \bigcup_{s \in S} \text{Run}^\mathcal{T}_{\text{fin}}(s)$. The length $|\sigma|$, or number of stages, of a finite run $\sigma = s_0 \ldots s_n$ is $n+1$ and $last(\sigma) = s_n$ denotes the last state of σ. With slight abuse of notation, we use $w(\sigma)$ to denote the weight of a finite run $\sigma = s_0 \ldots s_n$, i.e., $w(\sigma) = \sum_{i=0}^{n-1} w((s_i, s_{i+1}))$. Moreover, $w^*(s, s')$ denotes the minimum weight of a finite run from s to s'. Specifically, $w^*(s, s) = 0$ for every $s \in S$ and if there does not exist a run from s to s', then $w^*(s, s') = \infty$. For a set $S' \subseteq S$ we let $w^*(s, S') = \min_{s' \in S'} w^*(s, s')$. We say that a state s' and a set S' is reachable from s, iff $w^*(s, s') \neq \infty$ and $w^*(s, S') \neq \infty$, respectively.

Every run $\rho = s_0 s_1 \ldots \in \text{Run}^\mathcal{T}$, resp. $\sigma = s_0 \ldots s_n \in \text{Run}^\mathcal{T}_{\text{fin}}$, induces a word $z = L(s_0)L(s_1)\ldots \in (2^{AP})^\omega$, resp. $z = L(s_0)\ldots L(s_n) \in (2^{AP})^+$, over the power set of AP.

A cycle of the TS $\mathcal{T}$ is a finite run $\text{cyc} = c_0 \ldots c_m$ of $\mathcal{T}$ for which it holds that $(c_m, c_0) \in T$.

Definition 2: A subsystem of a TS $\mathcal{T} = (S, T, AP, L, w)$ is a TS $\mathcal{U} = (S_\mathcal{U}, T_\mathcal{U}, AP, L|_\mathcal{U}, w|_\mathcal{U})$, where $S_\mathcal{U} \subseteq S$ and $T_\mathcal{U} \subseteq T \cap (S_\mathcal{U} \times S_\mathcal{U})$. We use $L|_\mathcal{U}$ to denote the labeling function L restricted to the set $S_\mathcal{U}$. Similarly, we use $w|_\mathcal{U}$ with the obvious meaning. If the context is clear, we use L, w instead of $L|_\mathcal{U}, w|_\mathcal{U}$. A subsystem $\mathcal{U}$ of $\mathcal{T}$ is called strongly connected if for every pair of states $s, s' \in S_\mathcal{U}$, there exists a finite run $\sigma \in \text{Run}^\mathcal{U}_{\text{fin}}(s)$ such that $last(\sigma) = s'$. A strongly connected component (SCC) of $\mathcal{T}$ is a maximal strongly connected subsystem of $\mathcal{T}$. We use $\text{SCC}(\mathcal{T})$ to denote the set of all strongly connected components of $\mathcal{T}$.

Every state of a TS $\mathcal{T}$ belongs to at most one strongly connected component of $\mathcal{T}$. Hence, the cardinality of the set $\mathrm{SCC}(\mathcal{T})$ is bounded by the number of states of $\mathcal{T}$.

Definition 3: Let $\mathcal{T} = (S, T, AP, L, w)$ be a TS. A control strategy for $\mathcal{T}$ is a function $C : \mathrm{Run}_{\mathrm{fin}}^{\mathcal{T}} \to S$ such that for every $\sigma \in \mathrm{Run}_{\mathrm{fin}}^{\mathcal{T}}$, it holds that $(last(\sigma), C(\sigma)) \in T$.

A strategy C for which $C(\sigma_1) = C(\sigma_2)$, for all finite runs $\sigma_1, \sigma_2 \in \mathrm{Run}_{\mathrm{fin}}^{\mathcal{T}}$ with $last(\sigma_1) = last(\sigma_2)$, is called memoryless. In that case, C is a function $C : S \to S$. A strategy is called finite-memory if it can be defined as a tuple $C = (M, \mathsf{next}, \mathsf{trans}, \mathsf{start})$, where M is a finite set of modes, $\mathsf{trans} : M \times S \to M$ is a transition function, $\mathsf{next} : M \times S \to S$ selects a state of $\mathcal{T}$ to be visited next, and $\mathsf{start} : S \to M$ selects the starting mode. A strategy that is not finite-memory is called infinite-memory.

A run induced by C is a run $\rho_C = s_0 s_1 \ldots \in \mathrm{Run}^{\mathcal{T}}$ for which $s_{i+1} = C(\rho_C^{(i)})$ for every $i \geq 0$. For every $s \in S$, there is exactly one run induced by C that starts in s. A finite run induced by C is $\sigma_C \in \mathrm{Run}_{\mathrm{fin}}^{\mathcal{T}}$, which is a finite prefix of some ρ_C. Let C be a strategy, finite-memory or not, for a TS $\mathcal{T}$. For every state $s \in S$, the run $\rho_C \in \mathrm{Run}^{\mathcal{T}}(s)$ induced by C satisfies $\inf(\rho_C) \subseteq S_{\mathcal{U}}$ for some $\mathcal{U} \in \mathrm{SCC}(\mathcal{T})$ [1]. We say that C leads $\mathcal{T}$ from the state s to the SCC $\mathcal{U}$.

B. Linear Temporal Logic

Definition 4: Linear temporal logic (LTL) formulas over the 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 $a \in AP$ is an atomic proposition, $\neg$ and $\wedge$ are standard Boolean connectives, and $\mathbf{X}$ (*next*), $\mathbf{U}$ (until), $\mathbf{G}$ (always) and $\mathbf{F}$ (eventually) are temporal operators.

The semantics of LTL is defined over words over 2^{AP} [1], such as those generated by the runs of a TS $\mathcal{T}$. For example, a word $w \in (2^{AP})^{\omega}$ satisfies $\mathbf{G}\phi$ and $\mathbf{F}\phi$ if ϕ holds in w always and eventually, respectively. If the word induced by a run of $\mathcal{T}$ satisfies ϕ, we say that the run satisfies ϕ. We call ϕ satisfiable in $\mathcal{T}$ from $s \in S$ if there exists $\rho \in \mathrm{Run}^{\mathcal{T}}(s)$ that satisfies ϕ.

Having an initialized TS $\mathcal{T}$ and an LTL formula ϕ over AP, the formal synthesis problem aims to find a strategy C for $\mathcal{T}$ such that the run $\rho_C \in \mathrm{Run}^{\mathcal{T}}(s_{\mathrm{init}})$ induced by C satisfies ϕ. In that case we also say that the strategy C satisfies ϕ. The formal synthesis problem can be solved using principles from automata-based model checking [1]. Specifically, ϕ is translated to a Büchi automaton and the system combining the Büchi automaton and the TS $\mathcal{T}$ is analyzed.

Definition 5: A Büchi automaton (BA) is a tuple $\mathcal{B} = (Q, 2^{AP}, \delta, q_0, F)$, where Q is a nonempty finite set of states, 2^{AP} is the alphabet, $\delta \subseteq Q \times 2^{AP} \times Q$ is a transition relation such that for every $q \in Q$, $a \in 2^{AP}$, there exists $q' \in Q$ such that $(q, a, q') \in \delta$, $q_0 \in Q$ is the initial state, and $F \subseteq Q$ is a set of accepting states.

A run $q_0 q_1 \ldots Q^{\omega}$ of $\mathcal{B}$ is an infinite sequence such that for every $i \geq 0$ there exists $a_i \in 2^{AP}$ with $(q_i, a_i, q_{i+1}) \in \delta$. The word $a_0 a_1 \ldots \in (2^{AP})^{\omega}$ is called the word induced by the run $q_0 q_1 \ldots$. A run $q_0 q_1 \ldots$ of $\mathcal{B}$ is accepting if there exist infinitely many $i \geq 0$ such that $q_i \in F$.

For every LTL formula ϕ over AP, one can construct a Büchi automaton $\mathcal{B}_{\phi}$ such that the accepting runs are all and only words over 2^{AP} satisfying ϕ [22]. The size of such an automaton is in the worst case exponential in the size of the formula [23], but in practice the BA is often quite small and manageable. We refer readers to [23], [24] for algorithms and to online implementations such as [25], to translate an LTL formula to a BA.

Definition 6: Let $\mathcal{T} = (S, T, AP, L, w)$ be an initialized TS and $\mathcal{B} = (Q, 2^{AP}, \delta, q_0, F)$ be a Büchi automaton. The product $\mathcal{P}$ of $\mathcal{T}$ and $\mathcal{B}$ is a tuple $\mathcal{P} = (S_{\mathcal{P}}, T_{\mathcal{P}}, s_{\mathcal{P}init}, AP, L_{\mathcal{P}}, F_{\mathcal{P}}, w_{\mathcal{P}})$, where $S_{\mathcal{P}} = S \times Q$, $T_{\mathcal{P}} \subseteq S_{\mathcal{P}} \times S_{\mathcal{P}}$ is a transition relation such that for every $(s, q), (s', q') \in S_{\mathcal{P}}$ it holds that $((s, q), (s', q')) \in T_{\mathcal{P}}$ if and only if $(s, s') \in T$ and $(q, L(s), q') \in \delta$, $s_{\mathcal{P}init} = (s_{\mathrm{init}}, q_0)$ is the initial state, $L_{\mathcal{P}}((s, q)) = L(s)$ is a labeling function, $F_{\mathcal{P}} = S \times F$ is a set of accepting states, and $w_{\mathcal{P}}(((s, q), (s', q'))) = w((s, s'))$ is a weight function.

The product $\mathcal{P}$ can be viewed as an initialized TS with a set of accepting states. Therefore, we adopt the definitions of a run ρ, a finite run σ, its weight $w_{\mathcal{P}}(\sigma)$, and sets $\mathrm{Run}^{\mathcal{P}}((s, q))$, $\mathrm{Run}^{\mathcal{P}}$, $\mathrm{Run}_{\mathrm{fin}}^{\mathcal{P}}((s, q))$ and $\mathrm{Run}_{\mathrm{fin}}^{\mathcal{P}}$ from above. Similarly, a cycle cyc of $\mathcal{P}$, a strategy $C_{\mathcal{P}}$ for $\mathcal{P}$ and runs $\rho_{C_{\mathcal{P}}}, \sigma_{C_{\mathcal{P}}}$ induced by $C_{\mathcal{P}}$ are defined in the same way as for TS. We also adopt the definitions of a subsystem and a strongly connected component. On the other hand, $\mathcal{P}$ can be viewed as a weighted BA over the trivial alphabet with a labeling function, which gives us the definition of an accepting run of $\mathcal{P}$.

Every run $(s_0, q_0)(s_1, q_1)\ldots$ and finite run $(s_0, q_0)\ldots(s_n, q_n)$ of $\mathcal{P}$ projects to a run $s_0 s_1 \ldots$ and a finite run $s_0 \ldots s_n$ of $\mathcal{T}$, respectively. Vice versa, for every run $s_0 s_1 \ldots$ and finite run $s_0 \ldots s_n$ of $\mathcal{T}$, there exists a run $(s_0, q_0)(s_1, q_1)\ldots$ and finite run $(s_0, q_0)\ldots(s_n, q_n)$. Similarly, every strategy for $\mathcal{P}$ projects to a strategy for $\mathcal{T}$ and for every strategy for $\mathcal{T}$ there exists a strategy for $\mathcal{P}$ that projects to it. The projection of a finite-memory strategy for $\mathcal{P}$ is also finite-memory.

Definition 7: Let $\mathcal{P} = (S_{\mathcal{P}}, T_{\mathcal{P}}, s_{\mathcal{P}init}, AP, L_{\mathcal{P}}, F_{\mathcal{P}}, w_{\mathcal{P}})$ be the product of an initialized TS $\mathcal{T}$ and a BA $\mathcal{B}$. An accepting strongly connected component (ASCC) of $\mathcal{P}$ is an SCC $\mathcal{U} = (S_{\mathcal{U}}, T_{\mathcal{U}}, AP, L_{\mathcal{P}}, w_{\mathcal{P}})$ such that the set $S_{\mathcal{U}} \cap F_{\mathcal{P}}$ is nonempty and we refer to it as the set $F_{\mathcal{U}}$ of accepting states of $\mathcal{U}$. We use $\mathrm{ASCC}(\mathcal{P})$ to denote the set of all ASCCs of $\mathcal{P}$ that are reachable from the initial state $s_{\mathcal{P}init}$.

C. Markov Chain

Definition 8: A Markov chain (MC) is a tuple $\mathcal{M} = (G, P, p_{\mathrm{init}})$, where G is a nonempty finite set of states, $P : G \times G \to [0, 1]$ is a transition probability function such that $\sum_{g' \in G} P(g, g') = 1$ for all $g \in G$ and $p_{\mathrm{init}} : G \to [0, 1]$ is an initial distribution, i.e., $\sum_{g \in G} p_{\mathrm{init}}(g) = 1$.

A run of a Markov chain $\mathcal{M} = (G, P, p_{\mathrm{init}})$ is an infinite sequence $g_0 g_1 g_2 \ldots \in G^{\omega}$ such that for every $i \geq 0$ it holds $P(g_i, g_{i+1}) > 0$. A finite run of $\mathcal{M}$ is a finite prefix of a run of $\mathcal{M}$. A Markov chain is called strongly connected if for every pair $g, g' \in G$ of states there exists a finite run from g to g'. We call a Markov chain nontrivial if $|G| > 1$ implies that there exist $g, g' \in G$ such that $P(g, g') \in (0, 1)$.

In this paper, we only consider strongly connected nontrivial Markov chains due to reasons explained later in Rem. 2 and we interpret them as discrete stochastic processes [26], i.e., time series of values that involve probabilistic indeterminacy. We assume that the set of states $G = \{g_0, g_1, \dots g_n\}$ is an ordered finite set of nonnegative real numbers and we refer to G as the set of values. We use $\mathcal{M}(t)$ to denote the value (or state) of Markov chain $\mathcal{M}$ at time $t \in \mathbb{N}_0$. The function P can be represented as a square matrix $A_{\mathcal{M}} = \{A_{ij}\}$, where $A_{ij} = P(g_i, g_j)$.

Definition 9: The invariant distribution of a (strongly connected nontrivial) Markov chain $\mathcal{M} = (G, P, p_{\mathrm{init}})$, $G = \{g_0, \dots, g_n\}$, is a vector $\nu_{\mathcal{M}}$ of size $n+1$ such that for every $0 \le i \le n$ it holds $0 \le \nu_{\mathcal{M}}(i) \le 1$, $\sum_{i=0}^{n} \nu_{\mathcal{M}}(i) = 1$, and $\nu_{\mathcal{M}} \cdot A_{\mathcal{M}} = \nu_{\mathcal{M}}$.

Intuitively, $\nu_{\mathcal{M}}(i)$ is the probability of the Markov chain $\mathcal{M}$ being in state g_i, at any point of an execution. Note that the invariant distribution can be effectively computed using the above definition. The expected value of $\mathcal{M}$ is then

$$\mathcal{M}_E = \sum_{i=0}^{n} \nu_{\mathcal{M}}(i) \cdot g_i.$$

Assume that $\mathcal{M}(t) = g_i$ for some $t \ge 0$, $0 \le i \le n$, and let $k \ge 0$. The simulated expected value

$$\mathcal{M}_{\mathrm{sim}}(t, g_i, k) = \sum_{j=0}^{n} \left(A_{\mathcal{M}}^{k}\right)_{ij} \cdot g_j$$

is the expected value of $\mathcal{M}$ at time $t+k$ assuming that its value is g_i at time t.

III. Problem Formulation

Consider an initialized weighted transition system $\mathcal{T} = (S, T, AP, L, w)$. The weight $w((s, s'))$ represents the amount of time that the transition $(s, s') \in T$ takes and the system starts at time 0. We use t_n to denote the point in time after the nth transition of a run, i.e., initially $t_0 = 0$ and after a finite run $\sigma \in \mathrm{Run}_{\mathrm{fin}}^{\mathcal{T}}(s_{\mathrm{init}})$ of length $n+1$ the time is $t_n = w(\sigma)$.

We assume there is a dynamic *penalty* associated with every state of the transition system. The penalty in a state $s \in S$ is defined as a (strongly connected nontrivial) Markov chain $\mathcal{M}_s$. With slight abuse of notations, we use $g(s,t) = \mathcal{M}_s(t)$ to denote the value of the penalty in a state $s \in S$ at time $t \in \mathbb{N}_0$ and $g_E(s) = \mathcal{M}_{sE}$ is the expected value of the penalty in state s. Assuming that the penalty in state s is x at time $t \in \mathbb{N}_0$, $g_{\mathrm{sim}}(s, t, x, k) = \mathcal{M}_{s\mathrm{sim}}(t, x, k)$ is the simulated expected value of the penalty in state s at time $t+k$. We use $g_{\max}$ to denote the maximum possible value of a penalty over all states. Upon the visit of a state, the corresponding penalty is incurred. The visit of the state does not affect the penalty's value or dynamics.

In every execution of the transition system $\mathcal{T}$, the probabilistic choices during the evolution of all penalties are resolved in some particular way that we call *penalty profile*. A penalty profile Δ determines, for a particular execution of the TS $\mathcal{T}$, the value of penalty in every state at every time moment. The penalty profile that is being followed in an execution is not known to us. Nevertheless, we can compute the probability $\Pr(\Delta, k)$ that a penalty profile Δ will be followed in the first k time units of an execution based on the Markov chains $\mathcal{M}_s$, $s \in S$.

Motivated by robotic applications, where various sensors typically provide reasonable measurements only within certain range, we assume that the penalties are sensed only locally in close proximity from the current state. To be specific, we assume a *visibility range* $v \in \mathbb{N}$ is given. If the system is in a state $s \in S$ at time t, the penalty $g(s', t)$ of a state $s' \in S$ is observable if and only if $s' \in \mathrm{Vis}(s) = \{s' \in S \mid w^*(s, s') \le v\}$. The set $\mathrm{Vis}(s)$ is also called the set of states visible from s. We consider the penalties to be an integral part of the TS and thus, a strategy for the TS might consider in its decision procedure not only the sequence of states that have been visited in the past but also the penalties incurred and observed in the meantime. Therefore, the run induced by a strategy C for $\mathcal{T}$ may differ under different penalty profiles. We use $\rho_{C,\Delta}(s)$ to denote the run induced by a strategy C under a penalty profile Δ starting from a state $s \in S$.

The problem we consider in this paper combines the formal synthesis problem with long-term optimization of the expected amount of penalties incurred during the system's execution. We assume that the specification is given as an LTL formula ϕ of the form

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

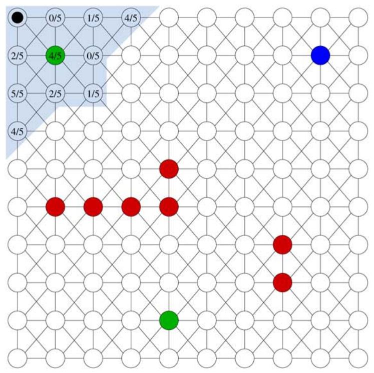
where φ is an LTL formula over AP and $\pi_{\mathrm{sur}} \in AP$. This formula requires that the system satisfies φ and surveys the states satisfying the property π_{sur} infinitely often. We say that every visit of a state in $S_{\mathrm{sur}} = \{s \in S \mid \pi_{\mathrm{sur}} \in L(s)\}$ completes a *surveillance cycle*. Specifically, starting from the initial state, the first visit of S_{sur} at $t > 0$ completes the first surveillance cycle of a run. Note that a surveillance cycle is not a cycle in the sense of the definition of a cycle of a TS in Section II. For a finite run σ such that $last(\sigma) \in S_{\mathrm{sur}}$, $\sharp(\sigma)$ denotes the number of complete surveillance cycles in σ, otherwise $\sharp(\sigma)$ is the number of complete surveillance cycles plus one.

Remark 1: The form in (1) does not restrict the expressiveness of LTL since every LTL formula φ over AP is equivalent to $\phi = \varphi \wedge \mathbf{GF}\pi_{\mathrm{sur}}$, where $\pi_{\mathrm{sur}} \in L(s)$ for every $s \in S$, i.e., it holds that a run of the TS $\mathcal{T}$ satisfies ϕ if and only if it satisfies φ. In this case, the long-term optimization objective minimizes the expected average penalty incurred per stage [27].

The long-term optimization objective is defined as follows. Let $V_{\mathcal{T},C} : S \to \mathbb{R}_0^+$ be a function such that $V_{\mathcal{T},C}(s)$ is the expected average cumulative penalty per surveillance cycle (APPC) incurred under a strategy C for $\mathcal{T}$ starting from a state $s \in S$:

$$V_{\mathcal{T},C}(s) = \limsup_{k \to \infty} \sum_{\Delta} \Pr(\Delta, k) \cdot \frac{\sum_{i=0}^{k} g\left(\rho_{C,\Delta}(i), w\left(\rho_{C,\Delta}^{(i)}\right)\right)}{\sharp\left(\rho_{C,\Delta}^{(k)}\right)} \tag{2}$$

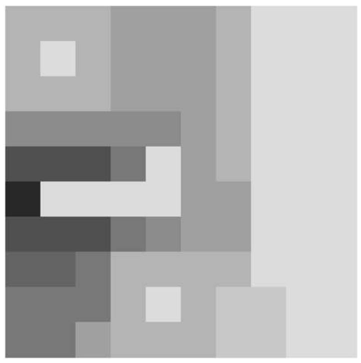
where $\rho_{C,\Delta} \in \mathrm{Run}^{\mathcal{T}}(s)$ is the run induced by C starting from s under a penalty profile Δ.



(a)

$$\begin{array}{c|cccccc} & 0 & 1/5 & 2/5 & 3/5 & 4/5 & 1 \\ \hline 0 & 0 & 1.0 & 0 & 0 & 0 & 0 \\ 1/5 & 0 & 0 & 1.0 & 0 & 0 & 0 \\ 2/5 & 0 & 0 & 0 & 1.0 & 0 & 0 \\ 3/5 & 0 & 0 & 0 & 0 & 1.0 & 0 \\ 4/5 & 0 & 0 & 0 & 0 & 0 & 1.0 \\ 1 & 1-p(s) & 0 & 0 & 0 & 0 & p(s) \end{array}$$

(b)



(c)

Fig. 1. (a) Transition system modeling the robot (black dot) motion in a partitioned environment. There are two delivery locations shown in green, a base shown in blue, and unsafe locations shown in red. (b) Transition matrix $A_{\mathcal{M}_s}$ of the Markov chain $\mathcal{M}_s$ that defines the penalty in a state s. (c) Graphical representation of the function $p : S \to (0, 1)$. The values range over the set $\{0.1, \ldots, 0.9\}$. Darker shades indicate higher values.

Problem 1: Let $\mathcal{T} = (S, T, AP, L, w)$ be an initialized TS, with penalties defined by (strongly connected nontrivial) Markov chains $\mathcal{M}_s$, $s \in S$. Let $v \in \mathbb{N}$ be a visibility range and ϕ an LTL formula over the set AP of the form in (1). Find a strategy C for $\mathcal{T}$ such that C satisfies ϕ and among all strategies satisfying ϕ, C minimizes the APPC value $V_{\mathcal{T},C}(s_{\text{init}})$ defined in (2).

Note that a strategy that solves Problem 1 is dependent on the penalty profiles and it defines an optimal control sequence for each penalty profile separately. However, due to the probabilistic nature of penalties, we are not able to predict their values in the future precisely, i.e., we are not able to determine the profile that is being followed during an execution. Hence, even if we were able to construct the solution to Problem 1, we do not have the means to use it. Therefore, we consider the following relaxed version of Problem 1. Consider strategies that are independent on the penalty profiles, i.e., $\rho_{C,\Delta}(s) = \rho_{C,\Delta'}(s)$ for any two penalty profiles Δ, Δ' and a state s. Note that such strategies may still consider the Markov chains defining the penalties in states of the TS and their expected values. For this type of strategies the value $V_{\mathcal{T},C}(s)$ for a state $s \in S$ from (2) can be computed easily as follows:

$$V_{\tau,C}(s) = \limsup_{k \to \infty} \frac{\sum_{i=0}^{k} g_E\left(\rho_C(i)\right)}{\sharp\left(\rho_C^{(k)}\right)}. \tag{3}$$

Problem 2: Let $\mathcal{T} = (S, T, AP, L, w)$ be an initialized TS, with penalties defined by (strongly connected nontrivial) Markov chains $\mathcal{M}_s$, $s \in S$. Let $v \in \mathbb{N}$ be a visibility range and ϕ an LTL formula over the set AP of the form in (1). Find a strategy C for $\mathcal{T}$ such that C is independent on penalty profiles, it satisfies ϕ, and among all strategies independent on penalty profiles and satisfying ϕ, it minimizes the APPC value $V_{\mathcal{T},C}(s_{\text{init}})$ defined in (3).

In Section IV-C, we propose an algorithm to design a strategy that solves Problem 2. Since the resulting strategy is independent on penalty profiles, it does not take advantage of the local sensing of penalties. It is computed in an offline manner and we refer to it as the offline strategy or offline control. While the offline strategy minimizes the APPC value among strategies satisfying the formula that are independent on penalty profiles, there may exist strategies that are dependent on penalty profiles and while satisfying the formula, provide lower APPC value than the offline control. We construct such a strategy in Section IV-D. Since the strategy is dependent on penalty profiles, it considers the penalties observed in real-time. This online control is constructed by locally improving the offline control according to the penalties observed from the current state of the TS and their simulation over the next h time units, where $h \in \mathbb{N}$ is a user-defined *planning horizon*. The online strategy is a heuristic, and we also suggest a method to construct a whole class of strategies with similar properties. In Section V, we evaluate all designed control strategies on illustrative case studies. All strategies synthesized in this work are infinite-memory in general, but can be implemented efficiently using simple technical improvements.

Example 1: Consider a robot whose motion in a grid-like partitioned environment is modeled by the transition system depicted in Fig. 1(a). The robot transports packages between two delivery locations, marked green in Fig. 1(a). The blue state marks the robot's base location. There is a transition between every two vertically, horizontally, and diagonally neighboring states. The weight of a horizontal and vertical transition is 2, for a diagonal transition it is 3. The Markov chain $\mathcal{M}_s$ defining the penalty in a state s has the set of values $G = \{0, 1/5, 2/5, 3/5, 4/5, 1\}$, the initial distribution is the uniform distribution over G and the transition matrix is of the form shown in Fig. 1(b). Intuitively, every penalty increases every time unit by 1/5 and always when the penalty is 1, in the next time unit the penalty remains 1 with nonzero probability $p(s)$ or it drops to 0 with nonzero probability $1 - p(s)$, where $p : S \to (0, 1)$ is a function over states of the system, defined in Fig. 1(c). The visibility range v is 6. For example, in Fig. 1(a) the set $\text{Vis}(s)$ of states visible from the state s, with corresponding penalties, is depicted as the blue-shaded area.

The mission for the robot is to transport packages between the two delivery locations (labeled with propositions a and b, respectively) and infinitely many times return to the base (labeled with c), while avoiding unsafe locations (labeled with u). At the same time, we wish to minimize the expected average cumulative penalty incurred per transport. To model this requirement, we add the property π_{sur} to the label set of both delivery locations. The corresponding LTL formula is

$$\begin{aligned}\mathbf{G}\,(a \Rightarrow \mathbf{X}(\neg a\,\mathbf{U}\,b)) \wedge \mathbf{G}\,(b \Rightarrow \mathbf{X}(\neg b\,\mathbf{U}\,a)) \\ \wedge\,\mathbf{GF}\,c \wedge \mathbf{G}(\neg u) \wedge \mathbf{GF}\,\pi_{\mathrm{sur}}. \qquad (4)\end{aligned}$$

IV. Solution

In this section, we present algorithms to construct the offline control and online control strategies. We prove their correctness, and discuss their complexity and usability.

A. Intuitive Description of the Approach

Both algorithms work with the product $\mathcal{P} = (S_\mathcal{P}, T_\mathcal{P}, s_{\mathcal{P}init}, AP, L_\mathcal{P}, F_\mathcal{P}, w_\mathcal{P})$ of the initialized TS $\mathcal{T}$ and a Büchi automaton $\mathcal{B}_\phi$ for the LTL formula ϕ. We map the penalties from $\mathcal{T}$ to $\mathcal{P}$ by defining $\mathcal{M}_{(s,q)} = \mathcal{M}_s$ for every state $(s,q) \in S_\mathcal{P}$ and $g((s,q),t) = g(s,t)$ for every $t \in \mathbb{N}_0$. We also adopt the visibility range v and the definition of the set $\mathrm{Vis}((s,q))$. We distinguish between control strategies for $\mathcal{P}$ that are dependent on penalty profiles and those that are not, i.e., between strategies for which the APPC value is computed directly using the definition in (2) and those for which the value can be computed easily using (3). The control strategies for $\mathcal{T}$ are computed as projections of control strategies for $\mathcal{P}$ with suitable properties.

In the offline algorithm presented in Section IV-C, we construct a strategy C for $\mathcal{T}$ that solves Problem 2 as a projection of a strategy $C_\mathcal{P}^{\mathsf{off}}$ for $\mathcal{P}$ that is independent on penalty profiles. The run induced by $C_\mathcal{P}^{\mathsf{off}}$ visits the set $F_\mathcal{P}$ infinitely many times and at the same time, the APPC value $V_{\mathcal{P},C_\mathcal{P}^{\mathsf{off}}}(s_{\mathcal{P}init})$ is minimal among all strategies for $\mathcal{P}$ that are independent on penalty profiles and visit the set $F_\mathcal{P}$ infinitely many times. To construct the strategy $C_\mathcal{P}^{\mathsf{off}}$, we leverage ideas from formal methods. Using the automata-based approach to model checking, one can construct a strategy $C_\mathcal{P}^{\mathsf{off},\phi}$ for $\mathcal{P}$ that visits at least one of the accepting states infinitely many times. On the other hand, using graph theory, we can design a strategy $C_\mathcal{P}^{\mathsf{off},V}$ that achieves the minimum APPC value among all strategies for $\mathcal{P}$ that are independent on penalty profiles and do not cause an immediate, unrepairable violation of ϕ, i.e., ϕ is satisfiable from every state of the run induced by $C_\mathcal{P}^{\mathsf{off},V}$. However, we would like to have a strategy $C_\mathcal{P}^{\mathsf{off}}$ satisfying both properties at the same time. To achieve that, we draw inspiration from a game theoretic technique in [28]. Intuitively, we combine two strategies $C_\mathcal{P}^{\mathsf{off},\phi}$ and $C_\mathcal{P}^{\mathsf{off},V}$ to create a new strategy $C_\mathcal{P}^{\mathsf{off}}$. The strategy $C_\mathcal{P}^{\mathsf{off}}$ is played in rounds, where each round consists of two phases. In the first phase, we play the strategy $C_\mathcal{P}^{\mathsf{off},\phi}$ until an accepting state is reached. We say that the aim is to achieve the mission subgoal. The second phase applies the strategy $C_\mathcal{P}^{\mathsf{off},V}$. The aim is to maintain the expected average cumulative penalty per surveillance cycle in the current round, and we refer to it as the average subgoal. The number of steps for which we apply $C_\mathcal{P}^{\mathsf{off},V}$ is computed individually every time we enter the second phase of a round. We prove the correctness of the proposed algorithm, discuss its complexity and usability of the resulting strategy for $\mathcal{T}$. Most importantly, the resulting offline control strategy is not a finite-memory strategy in general. Intuitively, we need to perform more and more steps in every round of the strategy. In Section IV-C, we discuss this matter in detail and suggest technical improvements that reduce the number of steps in every round and the usage of memory. The improvements result in a strategy that is equivalent to the offline strategy in the sense that it guarantees the satisfaction of the given LTL formula and has the same APPC value as the offline control strategy, but is no longer independent on penalty profiles.

The online algorithm described in Section IV-D constructs a strategy for $\mathcal{T}$ that also provably guarantees the satisfaction of given LTL formula but provides better or equal APPC than the offline strategy. Hence, in the context of Problem 1 it is a better solution than the offline control. It is however dependent on penalty profiles, so it cannot be considered as a solution to Problem 2. The online strategy is again computed as a projection of a suitable strategy $C_\mathcal{P}^{\mathsf{on}}$ for the product $\mathcal{P}$. We obtain $C_\mathcal{P}^{\mathsf{on}}$ by locally improving the strategy $C_\mathcal{P}^{\mathsf{off}}$ computed by the offline algorithm. Intuitively, we compare applying $C_\mathcal{P}^{\mathsf{off}}$ for several steps to reach a specific state or set of states of $\mathcal{P}$ to executing different local paths to reach the same state or set of states. We consider a finite set of finite runs leading to the state or set, containing the finite run induced by $C_\mathcal{P}^{\mathsf{off}}$. We then choose the one that is expected to minimize the average cumulative penalty per surveillance cycle incurred in the current round, taking into account the currently observed penalties within the visibility range, and apply the first transition of the chosen run. The process continues until the state, or set, is reached, and then it starts over again. For the same reasons as in the offline algorithm, the resulting strategy for $\mathcal{T}$ is an infinite-memory strategy. We again propose technical improvements to reduce memory usage and computational cost that result in a strategy for $\mathcal{T}$ that is equivalent to the online strategy, i.e., it guarantees the satisfaction of the given formula and provides the same APPC value. The online control strategy is a heuristic and we suggest a procedure to design a whole class of heuristic online strategies with similar properties.

B. Probability Measure

Let $C_\mathcal{P}$ be a strategy for $\mathcal{P}$ that is independent on penalty profiles and let $(s,q) \in S_\mathcal{P}$. Let $\sigma_{C_\mathcal{P}} \in \mathrm{Run}_{\mathrm{fin}}^{\mathcal{P}}((s,q))$ be a finite run induced by $C_\mathcal{P}$ starting from (s,q) and let $\tau \in [0, g_{\max}]^+$ be a sequence of length $|\sigma_{C_\mathcal{P}}|$ such that there exists a penalty profile for $\mathcal{P}$ for which the penalty

$$g\left(\sigma_{C_\mathcal{P}}(i), w_\mathcal{P}\left(\sigma_{C_\mathcal{P}}^{(i)}\right)\right) = \tau(i)$$

for every $0 \le i \le |\sigma_{C_\mathcal{P}}|$. We call $(\sigma_{C_\mathcal{P}}, \tau)$ a finite pair. Analogously, an infinite pair $(\rho_{C_\mathcal{P}}, \kappa)$ consists of the run $\rho_{C_\mathcal{P}} \in \mathrm{Run}^{\mathcal{P}}((s,q))$ induced by the strategy $C_\mathcal{P}$ and an infinite sequence $\kappa \in [0, g_{\max}]^\omega$ such that there exists a penalty profile for $\mathcal{P}$ for which the penalty

$$g\left(\sigma_{C_\mathcal{P}}(i), w_\mathcal{P}\left(\sigma_{C_\mathcal{P}}^{(i)}\right)\right) = \kappa(i)$$

for every $i \ge 0$. A cylinder set $Cyl((\sigma_{C_\mathcal{P}}, \tau))$ of a finite pair $(\sigma_{C_\mathcal{P}}, \tau)$ is the set of all infinite pairs $(\rho_{C_\mathcal{P}}, \kappa)$ for which τ is a prefix of κ.

Consider the σ-algebra generated by the set of cylinder sets of all finite pairs $(\sigma_{C_\mathcal{P}}, \tau)$, where $\sigma_{C_\mathcal{P}} \in \mathrm{Run}^{\mathcal{P}}_{\mathrm{fin}}((s,q))$. From classical concepts in probability theory [29], there exists a unique probability measure $\Pr^{\mathcal{P},C_\mathcal{P}}_{(s,q)}$ on the σ-algebra such that for a finite pair $(\sigma_{C_\mathcal{P}}, \tau)$

$$\Pr^{\mathcal{P},C_\mathcal{P}}_{(s,q)} \left(Cyl\left((\sigma_{C_\mathcal{P}}, \tau)\right)\right)$$

is the probability that the penalties incurred in the first $|\sigma_{C_\mathcal{P}}| + 1$ stages when applying the strategy $C_\mathcal{P}$ in $\mathcal{P}$ from the state (s,q) are given by the sequence τ, i.e.,

$$g\left(\sigma_{C_\mathcal{P}}(i), w_\mathcal{P}\left(\sigma^{(i)}_{C_\mathcal{P}}\right)\right) = \tau(i)$$

for every $0 \leq i \leq |\sigma_{C_\mathcal{P}}|$. This probability can be computed based on the Markov chains $\mathcal{M}_{(s,q)}$, $(s,q) \in S_\mathcal{P}$ and it is equal to the sum of probabilities $\Pr(\Delta, w_\mathcal{P}(\sigma_{C_\mathcal{P}}))$ over all penalty profiles Δ under which the above equation is satisfied. For a set X of infinite pairs, an element of the above σ-algebra, the probability $\Pr^{\mathcal{P},C_\mathcal{P}}_{(s,q)}(X)$ is the probability that under $C_\mathcal{P}$ starting from (s,q) the infinite sequence of penalties incurred in the visited states is κ for some $(\rho_{C_\mathcal{P}}, \kappa) \in X$.

C. Offline Control

In this section, we construct a strategy $C^{\mathsf{off}}_\mathcal{P}$ for $\mathcal{P}$ and prove that its projection the offline control strategy C for $\mathcal{T}$ solves Problem 2. All strategies in this subsection are independent on penalty profiles, i.e., their APPC value can be computed using (3).

The strategy $C^{\mathsf{off}}_\mathcal{P}$ must visit the set $F_\mathcal{P}$ infinitely many times and hence, it must lead from $s_{\mathcal{P}init}$ to an ASCC. If the set $\mathrm{ASCC}(\mathcal{P})$ is empty, the formula ϕ cannot be satisfied in $\mathcal{T}$ and our algorithm terminates. Otherwise, for $\mathcal{U} \in \mathrm{ASCC}(\mathcal{P})$, we denote $V^*_\mathcal{U}((s,q))$ the minimum expected average cumulative penalty per surveillance cycle that can be achieved in $\mathcal{U}$ by a strategy independent on penalty profiles starting from $(s,q) \in S_\mathcal{U}$. Since $\mathcal{U}$ is strongly connected, this value is the same for all states in $S_\mathcal{U}$ and is denoted by $V^*_\mathcal{U}$. It is associated with a cycle $\mathsf{cyc}^V_\mathcal{U} = c_0 \ldots c_m$ of $\mathcal{U}$ witnessing the value. We describe the algorithm to compute $V^*_\mathcal{U}$ and $\mathsf{cyc}^V_\mathcal{U}$ for $\mathcal{U}$ in the proof of Th. 1. In the remainder of this section, $\mathcal{U} = (S_\mathcal{U}, T_\mathcal{U}, AP, L_\mathcal{P}, w_\mathcal{P})$ is the ASCC of the product that minimizes $V^*_\mathcal{U}$ and $\mathsf{cyc}^V_\mathcal{U} = c_0 \ldots c_m$ is the corresponding cycle.

Now, we design the strategies $C^{\mathsf{off},\phi}_\mathcal{P}$ and $C^{\mathsf{off},V}_\mathcal{P}$ that are then combined to create the strategy $C^{\mathsf{off}}_\mathcal{P}$. The strategy $C^{\mathsf{off},\phi}_\mathcal{P}$ is a memoryless strategy that from every $(s,q) \in S_\mathcal{P} \setminus F_\mathcal{U}$ that can reach the set $F_\mathcal{U}$, follows one of the finite runs with the minimum weight from (s,q) to $F_\mathcal{U}$. Formally, for every $(s,q) \in S_\mathcal{P} \setminus F_\mathcal{U}$ with $w^*_\mathcal{P}((s,q), F_\mathcal{U}) < \infty$, we have $C^{\mathsf{off},\phi}_\mathcal{P}((s,q)) = (s',q')$, where

$$w_\mathcal{P}\left((s,q),(s',q')\right) = w^*_\mathcal{P}\left((s,q), F_\mathcal{U}\right) - w^*_\mathcal{P}\left((s',q'), F_\mathcal{U}\right).$$

The strategy $C^{\mathsf{off},V}_\mathcal{P}$ is a memoryless strategy given by the cycle $\mathsf{cyc}^V_\mathcal{U} = c_0 \ldots c_m$. Similarly to the above strategy $C^{\mathsf{off},\phi}_\mathcal{P}$, for a state $(s,q) \in S_\mathcal{P} \setminus \mathsf{cyc}^V_\mathcal{U}$ with $w^*_\mathcal{P}((s,q), \mathsf{cyc}^V_\mathcal{U}) < \infty$, the strategy $C^{\mathsf{off},V}_\mathcal{P}$ follows one of the finite runs with the minimum weight to $\mathsf{cyc}^V_\mathcal{U}$. For a state $c_i \in \mathsf{cyc}^V_\mathcal{U}$, it holds $C^{\mathsf{off},V}_\mathcal{P}(c_i) = c_{i+1 \bmod (m+1)}$. The strategy $C^{\mathsf{off},V}_\mathcal{P}$ has the following property.

Proposition 1: For the strategy $C^{\mathsf{off},V}_\mathcal{P}$ and every state $(s,q) \in S_\mathcal{U}$ it holds that for every $\epsilon > 0$, there exists $j(\epsilon) \in \mathbb{N}$ such that if $C^{\mathsf{off},V}_\mathcal{P}$ is followed from the state (s,q) until at least $j(\epsilon)$ surveillance cycles are completed, then the average cumulative penalty per surveillance cycle incurred in the performed finite run is at most $V^*_\mathcal{U} + \epsilon$ with probability at least $1 - \epsilon$. Formally,

$$\lim_{k\to\infty} \Pr^{\mathcal{U},C^{\mathsf{off},V}_\mathcal{P}}_{(s,q)} \left(\frac{\sum_{i=0}^{k} g\left(\rho_{C^{\mathsf{off},V}_\mathcal{P}}(i), w_\mathcal{P}\left(\rho^{(i)}_{C^{\mathsf{off},V}_\mathcal{P}}\right)\right)}{\sharp\left(\rho^{(k)}_{C^{\mathsf{off},V}_\mathcal{P}}\right)} \leq V^*_\mathcal{U} \right) = 1. \tag{5}$$

Proof: It holds that the product $\mathcal{P}$ with penalties defined as MCs can be translated into a Markov decision process (MDP) with static penalties. Together with the fact that the cycle $\mathsf{cyc}^*_\mathcal{U}$ provides the minimum APPC value in the ASCC $\mathcal{U}$, it implies that (5) is equivalent to the property of MDPs proved in [28] regarding the minimum expected penalty per stage. □

Remark 2: Here we explain why we require that the Markov chains defining the penalties of the TS $\mathcal{T}$ be nontrivial. Assume there exists a state $(s,q) \in S_\mathcal{P}$ with a trivial Markov chain $\mathcal{M}_{(s,q)}$, i.e., the penalty in (s,q) evolves deterministically, not probabilistically. If we visit (s,q) infinitely many times in different (not necessarily consequent) points in time, the expected average penalty incurred in (s,q) might differ from $g_E((s,q))$. That can cause violation of Prop. 1.

Finally, we are ready to define the strategy $C^{\mathsf{off}}_\mathcal{P}$. It is played in rounds, where each round consists of two phases, one for each subgoal. The first round starts at the beginning of the execution of the system in the initial state $s_{\mathcal{P}init}$ of $\mathcal{P}$. Let i be the current round. In the first phase of the round the strategy $C^{\mathsf{off},\phi}_\mathcal{P}$ is applied until an accepting state of the ASCC $\mathcal{U}$ is reached. We use k_i to denote the number of steps we played the strategy $C^{\mathsf{off},\phi}_\mathcal{P}$ in round i. Once the mission subgoal is fulfilled, the average subgoal becomes the current subgoal. In this phase, we play the strategy $C^{\mathsf{off},V}_\mathcal{P}$ until the number of completed surveillance cycles in the second phase of the current round is $l_i = \max\{j(1/i), i \cdot (k_i + |S_\mathcal{U}|) \cdot g_{\max}\}$, where $j(1/i)$ is from Prop. 1.

Theorem 1: The offline control strategy C that results from projecting the strategy $C^{\mathsf{off}}_\mathcal{P}$ from $\mathcal{P}$ to $\mathcal{T}$ solves Problem 2 and the APPC value $V_{\mathcal{T},C}(s_{\mathrm{init}}) = V_{\mathcal{P},C^{\mathsf{off}}_\mathcal{P}}(s_{\mathcal{P}init}) = V^*_\mathcal{U}$.

Proof: To prove that the offline strategy C satisfies the LTL formula ϕ, we show that $C^{\mathsf{off}}_\mathcal{P}$ guarantees infinite number of visits of accepting states. Since the ASCC $\mathcal{U}$ is reachable from the initial state $s_{\mathcal{P}init}$ and from the construction of $C^{\mathsf{off},\phi}_\mathcal{P}$, it holds that every round of the strategy $C^{\mathsf{off}}_\mathcal{P}$ finishes after a finite number of steps and in every round an accepting state is visited.

To prove that the offline control strategy minimizes the APPC value among all strategies that satisfy the LTL formula ϕ, we first present the algorithm to compute the minimum APPC value $V^*_\mathcal{U}$ that can be achieved in an ASCC $\mathcal{U}$ and a cycle $\mathsf{cyc}^V_\mathcal{U}$ of $\mathcal{U}$ witnessing the value. The idea is to reduce $\mathcal{U}$ to a TS $\overline{\mathcal{U}}$ that contains only the states labeled with the surveillance

```
Input: an ASCC $\mathcal{U} = (S_\mathcal{U}, T_\mathcal{U}, AP, L_\mathcal{P}, w_\mathcal{P})$ of $\mathcal{P}$
Output: a TS $\overline{\mathcal{U}} = (S_{\mathcal{U}\mathrm{sur}}, T_{\overline{\mathcal{U}}}, AP, L_\mathcal{P}, w_{\overline{\mathcal{U}}})$ and a function $\mathrm{run}\colon T_{\overline{\mathcal{U}}} \to \mathrm{Run}^{\mathcal{U}}_{\mathrm{fin}}$
 1: let $X = (S_X, T_X, AP, L_\mathcal{P}, w_X)$ be a TS equal to $\mathcal{U}$ and let $\mathrm{run}_X\colon T_X \to \mathrm{Run}^{\mathcal{U}}_{\mathrm{fin}}$ be a function such that $\mathrm{run}_X((u,u')) = u$ for every $(u,u') \in T_X$
 2: while $S_X \setminus S_{\mathcal{U}\mathrm{sur}} \neq \emptyset$ do
 3:   let $u \in S_X \setminus S_{\mathcal{U}\mathrm{sur}}$
 4:   for all $u_1, u_2 \in S_X, u_1 \neq u, u_2 \neq u, (u_1,u),(u,u_2) \in T_X$ do
 5:     if $(u_1,u_2) \notin T_X$ then
 6:       add $(u_1,u_2)$ to $T_X$
 7:       $\mathrm{run}_X((u_1,u_2)) := \mathrm{run}_X((u_1,u)).\mathrm{run}_X((u,u_2))$
 8:       $w_X((u_1,u_2)) := \sum g_E((u_1,u_2))$
 9:     else
10:       if $\sum g_E(\mathrm{run}_X((u_1,u_2))) \geq \sum g_E(\mathrm{run}_X((u_1,u)).\mathrm{run}_X((u,u_2)))$ then
11:         $\mathrm{run}_X((u_1,u_2)) := \mathrm{run}_X((u_1,u)).\mathrm{run}_X((u,u_2))$
12:         $w_X((u_1,u_2)) := \sum g_E((u_1,u_2))$
13:       end if
14:     end if
15:   end for
16:   remove $u$ from $S_X$, and all adjacent transitions from $T_X$
17: end while
18: return $X$ and $\mathrm{run}_X$
```

Fig. 2. Algorithm for the reduction of an ASCC. We use . to denote the concatenation of two finite sequences and $\sum g_E(x)$ is the sum of expected penalties $g_E(x(i))$ for every state $x(i)$ of a finite run x.

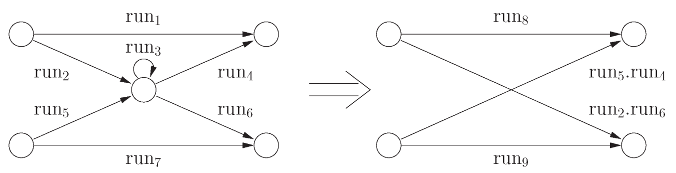


Fig. 3. Elimination of one state of an ASCC during the algorithm in Fig. 2. The finite run run_8 is the one of the runs run_1 and $\mathrm{run}_2.\mathrm{run}_4$ that minimizes the sum of expected penalties in the states of the run. Similarly, run_9 is one of the runs run_7 and $\mathrm{run}_5.\mathrm{run}_6$.

proposition π_{sur} and then apply Karp's algorithm [30] that finds a cycle with minimum value per edge also called the minimum mean cycle for a directed graph with values on edges. The value $V^*_\mathcal{U}$ and cycle $\mathsf{cyc}^V_\mathcal{U}$ are synthesized from the minimum mean cycle of $\overline{\mathcal{U}}$.

In Fig. 2, we present the algorithm that given an ASCC $\mathcal{U} = (S_\mathcal{U}, T_\mathcal{U}, AP, L_\mathcal{P}, w_\mathcal{P})$ of $\mathcal{P}$ returns a TS $\overline{\mathcal{U}} = (S_{\mathcal{U}\mathrm{sur}}, T_{\overline{\mathcal{U}}}, AP, L_\mathcal{P}, w_{\overline{\mathcal{U}}})$ and a function $\mathrm{run}: T_{\overline{\mathcal{U}}} \to \mathrm{Run}^{\mathcal{U}}_{\mathrm{fin}}$ with the following properties. For simplicity, we use singletons such as u, u_i to denote the states of $\mathcal{P}$. For the TS $\overline{\mathcal{U}}$ it holds that $(u,u') \in T_{\overline{\mathcal{U}}}$ if and only if there exists a finite run in $\mathcal{U}$ from $u \in S_{\mathcal{U}\mathrm{sur}}$ to $u' \in S_{\mathcal{U}\mathrm{sur}}$ with one surveillance cycle, i.e., between u and u' no state labeled with π_{sur} is visited. Moreover, the run $\mathrm{run}((u,u')) = u_0 \ldots u_n$ is such that $u = u_0$ and $\sigma = u_0 \ldots u_n u'$ is the finite run in $\mathcal{U}$ from u to u' with one surveillance cycle that minimizes the sum of expected penalties received during σ among all finite runs in $\mathcal{U}$ from u to u' with one surveillance cycle. The algorithm in Fig. 2 builds $\overline{\mathcal{U}}$ and the function run by eliminating the states from $S_\mathcal{U} \setminus S_{\mathcal{U}\mathrm{sur}}$ one by one, in arbitrary order. Fig. 3 demonstrates elimination of one such state on an illustrative example.

We apply the Karp's algorithm to the oriented graph with vertices $S_{\mathcal{U}\mathrm{sur}}$, edges $T_{\overline{\mathcal{U}}}$ and values on edges $w_{\overline{\mathcal{U}}}$. Let $\mathsf{cyc}_{\overline{\mathcal{U}}} = u_0 \ldots u_m$ be the minimum mean cycle of this graph. We have

$$V^*_\mathcal{U} = \frac{1}{m+1} \sum_{i=0}^{m} g_E\left(\mathrm{run}\left((u_i, u_{i+1 \bmod (m+1)})\right)\right)$$

$$\mathsf{cyc}^V_\mathcal{U} = \mathrm{run}((u_0,u_1)) . \ \ldots \ .\mathrm{run}((u_{m-1},u_m)) .\mathrm{run}((u_m,u_0)) .$$

It can be easily shown that all states of the cycle $\mathsf{cyc}^V_\mathcal{U}$ are distinct. It follows that

$$\frac{1}{\left|\mathsf{cyc}^V_\mathcal{U} \cap S_{\mathcal{U}\mathrm{sur}}\right|} \sum_{i=0}^{m} g_E(c_i) = V^*_\mathcal{U}.$$

When the APPC value and the corresponding cycle is computed for every ASCC of $\mathcal{P}$, we choose the ASCC that minimizes the APPC value. We denote this ASCC $\mathcal{U} = (S_\mathcal{U}, T_\mathcal{U}, AP, L_\mathcal{P}, w_\mathcal{P})$ and $\mathsf{cyc}^V_\mathcal{U} = c_0 \ldots c_m$. Every strategy solving Problem 2 must achieve APPC value $V^*_\mathcal{U}$.

Since the offline strategy C is a projection of the strategy $C^{\mathsf{off}}_\mathcal{P}$, we have $V_{\mathcal{T},C}(s_{\mathrm{init}}) = V_{\mathcal{P},C^{\mathsf{off}}_\mathcal{P}}(s_{\mathcal{P}init})$. To show that $V_{\mathcal{P},C^{\mathsf{off}}_\mathcal{P}}(s_{\mathcal{P}init}) = V^*_\mathcal{U}$, let $\epsilon_i = 1/i$ for round i. From Prop. 1 and the fact that $l_i = \max\{j(1/i), i \cdot (k_i + |S_\mathcal{U}|) \cdot g_{\max}\}$ it follows that the average penalty per surveillance cycle in ith round after its completion is at most

$$\begin{aligned} &\frac{k_i \cdot g_{\max} + |S_\mathcal{U}| \cdot g_{\max} + l_i\left(V^*_\mathcal{U} + \epsilon_i\right)}{l_i} \\ &\quad \leq V^*_\mathcal{U} + \epsilon_i + \frac{1}{i} \qquad\qquad (l_i \geq i \cdot (k_i + |S_\mathcal{U}|) \cdot g_{\max}) \\ &\quad = V^*_\mathcal{U} + \frac{2}{i} \end{aligned}$$

with probability at least $1 - (1/i)$. Therefore, in the limit, the average cumulative penalty per surveillance cycle $V^*_\mathcal{U}$ with probability 1, independently on penalty profile. □

Complexity: The size of a BA for an LTL formula ϕ is in $2^{\mathcal{O}(|\phi|)}$, where $|\phi|$ is the size of ϕ [23]. However, the actual size of the BA is in practice often quite small. The size of the product $\mathcal{P}$ is in $\mathcal{O}(|S| \cdot 2^{\mathcal{O}(|\phi|)})$. To compute the minimum weights $w^*((s,q),(s',q'))$ between every two states of $\mathcal{P}$ we use Floyd–Warshall algorithm taking $\mathcal{O}(|S_\mathcal{P}|^3)$ time. Tarjan's algorithm [31] is used to compute the set $\mathrm{SCC}(\mathcal{P})$ in $\mathcal{O}(|S_\mathcal{P}| + |T_\mathcal{P}|)$ time. The reduction of an ASCC $\mathcal{U}$ can be computed in $\mathcal{O}(|S_\mathcal{U}| \cdot |T_\mathcal{U}|^2)$ time. The Karp's algorithm [30] finds the optimal APPC value and corresponding cycle in $\mathcal{O}(|S_{\overline{\mathcal{U}}}| \cdot |T_{\overline{\mathcal{U}}}|)$ time. The main pitfall of the algorithm is to compute the number $j(1/i)$ of surveillance cycles needed in the second phase of the current round i according to Prop. 1. Intuitively, we need to consider the finite run $\sigma_{C^{\mathsf{off},V}_\mathcal{P},k}$ induced by the strategy $C^{\mathsf{off},V}_\mathcal{P}$ from the current state that contains $k = 1$ surveillance cycles, and compute the sum of probabilities $\Pr^{\mathcal{P},C_\mathcal{P}}_{(s,q)}(Cyl((\sigma_{C^{\mathsf{off},V}_\mathcal{P},k}, \tau)))$ for every τ with the average cumulative penalty per surveillance cycle less or equal to $V^*_\mathcal{U} + (2/i)$. If the total probability is at least $1 - (1/i)$, we set $j(1/i) = k$, otherwise we increase k and repeat the process. For every k, there exist up to $g_{\max}$ to the power of $|\sigma_{C^{\mathsf{off},V}_\mathcal{P},k}|$ sequences τ. This issue can be partially overcome using the rule presented below.

Usability: The strategy $C^{\mathsf{off}}_\mathcal{P}$ is not a finite-memory strategy in general. The reason is that the number of surveillance cycles that we need to perform in the second phase of round i is increasing with i. Note that in the special case when there exists a cycle $\mathsf{cyc}^V_{\mathcal{U}}$ of the SCC $\mathcal{U}$ corresponding to $V^*_{\mathcal{U}}$ that contains an accepting state, the memoryless strategy $C^{\mathsf{off},V}_{\mathcal{P}}$ for the average subgoal maps to a strategy of $\mathcal{T}$ solving Problem 2, which is therefore in the worst case finite-memory. To improve the memory usage we suggest the following technical improvements. Let $\overline{C^{\mathsf{off}}_{\mathcal{P}}}$ be the strategy for $\mathcal{P}$ that results from applying the following rule to the strategy $C^{\mathsf{off}}_{\mathcal{P}}$. Let i be the current round and k_i the number of steps in the first phase of the round. In the second phase we proceed as follows. After completion of every surveillance cycle, check whether the average penalty per surveillance cycle incurred in the current round of the execution is above $V^*_{\mathcal{U}} + (2/i)$, for the significance of this value see the proof of Th. 1. If yes, continue with the second phase of round i, otherwise start new round $i+1$. Also, avoid performing the expensive computation of the value $j(1/i)$ until it is necessary, i.e., only compute the value once the number of surveillance cycles performed in the second phase of the round i is $i \cdot (k_i + |S_{\mathcal{U}}|) \cdot g_{\max}$ and the average penalty per surveillance cycle in the round i is still above $V^*_{\mathcal{U}} + (2/i)$. Note that the strategy $\overline{C^{\mathsf{off}}_{\mathcal{P}}}$ is dependent on penalty profiles but it is equivalent to the strategy $C^{\mathsf{off}}_{\mathcal{P}}$ in the meaning that it provably guarantees infinite number of visits to the set $F_{\mathcal{P}}$ of accepting states and the APPC value of $\overline{C^{\mathsf{off}}_{\mathcal{P}}}$ is equal to the APPC value of $C^{\mathsf{off}}_{\mathcal{P}}$, i.e., it is $V^*_{\mathcal{U}}$. Formally, the strategy $\overline{C^{\mathsf{off}}_{\mathcal{P}}}$ may still require infinite memory. However, in our simulations in Section V we demonstrate that the memory usage and the amount of computation performed while using the strategy $\overline{C^{\mathsf{off}}_{\mathcal{P}}}$ is significantly decreased comparing to the strategy $C^{\mathsf{off}}_{\mathcal{P}}$. More specifically, the number of surveillance cycles performed in the second phase of each round drops dramatically and the value $j(1/i)$ for round i needs to be computed only rarely, if ever.

Remark 3: By considering only the expected values of penalties in states of the TS, the penalties can be seen as static in our offline approach. This makes Problem 2 related to the problem formulated in [18] for systems with static costs. In [18], the optimality is achieved by persistently increasing the number of visits to states under surveillance that leads to neglecting the remaining part of the LTL specification over time. The authors in [18] disregard such a strategy as undesirable. In our case, the optimal strategy $C^{\mathsf{off}}_{\mathcal{P}}$ performs only as many surveillance cycles in every round as are needed for the expected average penalty per surveillance cycle to be close enough to the optimal value $V^*_{\mathcal{U}}$ with probability 1, see Prop. 1 and the proof of Th. 1. This allows us to efficiently implement $C^{\mathsf{off}}_{\mathcal{P}}$ using feedback, see the discussion on usability above. We demonstrate in Section V that unlike the one in [18], our optimal strategy does not lead to the undesirable, ever-increasing number of visits of surveillance states.

Example 2: For the delivery system from Ex. 1, the Büchi automaton generated for the LTL formula in (4) using [25] has 16 states. The product $\mathcal{P}$ of the transition system and the automaton contains 2 ASCCs and the chosen, optimal ASCC $\mathcal{U}$ has 568 states. The projection of a cycle $\mathsf{cyc}^V_{\mathcal{U}}$ providing the minimum expected average cumulative penalty per surveillance cycle is depicted in magenta in Fig. 4 and the APPC value associated with the cycle is $V^*_{\mathcal{U}} = 4.35$.

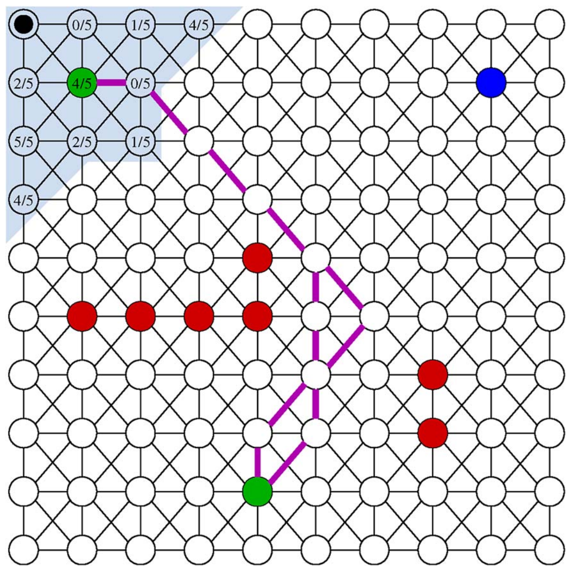


Fig. 4. Transition system for the delivery system example. The projection of an optimal APPC cycle is shown in magenta.

The offline control has the following structure. In the first phase of the first round, the aim of the strategy $C^{\mathsf{off}}_{\mathcal{P}}$ for the product is to reach an accepting state of the ASCC $\mathcal{U}$ using a finite run of the minimum possible weight. When projected to the TS, the robot starts from the base location and moves to the top delivery location using a finite run of the minimum possible weight. The first phase is completed one step after the visit of the delivery location, when the product reaches an accepting state, in total of $k_1 = 8$ steps. The control proceeds to the second phase of the first round. In the product, we first reach the optimal APPC cycle $\mathsf{cyc}^V_{\mathcal{U}}$ as fast as possible and then follow the cycle until $l_1 = \max\{576, j(1)\}$ surveillance cycles are completed. At the beginning of the second phase, even though the robot might be in a state that lies on the cycle shown in magenta in Fig. 4, the product is not yet on the cycle $\mathsf{cyc}^V_{\mathcal{U}}$. Therefore, in the TS the control does not yet follow the cycle depicted in Fig. 4. In the product, the closest state of the optimal APPC cycle that can be reached is the one that projects to the bottom delivery location. That means, the robot follows one of the shortest finite runs from the top to the bottom delivery location. The round then proceeds with l_1 alternative visits of the two delivery locations by following the cycle shown in magenta in Fig. 4. Once completed, the control proceeds to the second round. Every round $i \geq 2$ starts from either the top or bottom delivery location. In the first phase of the round, the robot first moves from the top, or bottom, delivery location to the base location and then continues delivering the packages by moving to the bottom, or top, delivery location, respectively. In both cases the first phase of the round ends after 15 steps, i.e., $l_i = 15$ for every $i \geq 2$. In the second phase, the robot keeps delivering packages between the top and the bottom delivery locations until the number of visits of the two locations is $l_i = \max\{i \cdot 583, j(1/i)\}$.

Note that the robot follows a predetermined sequence of transitions given by the strategy $C^{\mathsf{off}}_{\mathcal{P}}$, without considering the penalties incurred or observed in real-time. Also, the robot visits the base location only during the first phase of rounds and as the number l_i grows very fast with i, the time between consecutive visits to the base grows rapidly. By applying the rule from the above discussion on the usability of the offline control, we can visit the base more often by monitoring the amount of penalties incurred in the current round and decreasing the number of performed surveillance cycles. As we demonstrate in Section V-B, the rule dramatically improves the visit rate of the base while not affecting the convergence to the optimal value $V_{\mathcal{U}}^*$.

D. Online Control

The online algorithm locally improves strategy $C_{\mathcal{P}}^{\mathsf{off}}$ designed in the previous section according to the values of penalties observed from the current state and their simulation in the next h time units, where $h \in \mathbb{N}$ is a user-defined planning horizon. The resulting strategy $C_{\mathcal{P}}^{\mathsf{on}}$ as well as its projection to the TS $\mathcal{T}$, the online control strategy, are therefore dependent on penalty profiles. The strategy $C_{\mathcal{P}}^{\mathsf{on}}$ is again played in rounds. In each step of the strategy $C_{\mathcal{P}}^{\mathsf{on}}$, we consider a finite set of finite runs starting from the current state, choose one according to an optimization function, and apply its first transition.

In this section, we use the following notation. We use singletons such as u, u_i to denote the states of $\mathcal{P}$. Let $\sigma_{\mathrm{all}} \in \mathrm{Run}_{\mathrm{fin}}^{\mathcal{P}}(s_{\mathcal{P}init})$ denote the finite run executed by $\mathcal{P}$ so far. Let i be the current round of strategy $C_{\mathcal{P}}^{\mathsf{on}}$ and $\sigma_i = u_{i,0} \ldots u_{i,k}$ the finite run executed so far in this round, i.e., $u_{i,k}$ is the current state of $\mathcal{P}$. We use $t_{i,0}, \ldots, t_{i,k}$ to denote the points in time when the states $u_{i,0}, \ldots, u_{i,k}$ were visited, respectively.

The optimization function $f : \mathrm{Run}_{\mathrm{fin}}^{\mathcal{P}}(u_{i,k}) \to \mathbb{R}_0^+$ assigns every finite run $\sigma = u_0 \ldots u_n$ starting from the current state value $f(\sigma)$ that is the expected average cumulative penalty per surveillance cycle that would be incurred in round i, if run σ was to be executed next, i.e.,

$$f(\sigma) = \frac{\sum\limits_{j=0}^{k} g(u_{i,j}, t_{i,j}) + \sum\limits_{j=1}^{n} \overline{g}\left(u_j, t_{i,k} + w_{\mathcal{P}}\left(\sigma^{(j)}\right)\right)}{\sharp\left(\sigma_i.\sigma(1). \ \ldots \ .last(\sigma)\right)} \tag{6}$$

where

$$\overline{g}\Big(u_j, t_{i,k} + w_{\mathcal{P}}\Big(\sigma^{(j)}\Big)\Big) = \begin{cases} g_{\mathrm{sim}}\big(u_j, t_{i,k}, g(u_j, t_{i,k}), w_{\mathcal{P}}\big(\sigma^{(j)}\big)\big) \\ \quad \text{if } u_j \in \mathrm{Vis}(u_{i,k}), w_{\mathcal{P}}\big(\sigma^{(j)}\big) \le h \\ g_E(u_j) \\ \quad \text{otherwise.} \end{cases}$$

Intuitively, if the penalty in state u_j is visible in the current time moment and u_j would have been visited within the next h time units in run σ, the value $\overline{g}(u_j, t_{i,k} + w_{\mathcal{P}}(\sigma^{(j)}))$ refers to the simulated expected penalty in u_j at the time of its visit, as defined in Section II-C. Otherwise, we do not simulate the penalty over time and consider only its expected value $g_E(u_j)$.

For a set of states $X \subseteq S_{\mathcal{P}}$, we define a *shortening indicator function* $I_X : T_{\mathcal{P}} \to \{0, 1\}$ such that for $t_{\mathcal{P}} = ((s_1, q_1), (s_2, q_2)) \in T_{\mathcal{P}}$, we have

$$I_X(t_{\mathcal{P}}) = \begin{cases} 1 & \text{if } w_{\mathcal{P}}^*\left((s_1, q_1), X\right) > w_{\mathcal{P}}^*\left((s_2, q_2), X\right), \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

In words, the indicator has value 1 if the transition leads strictly closer to X, and 0 otherwise.

Now we are ready to formally define the strategy $C_{\mathcal{P}}^{\mathsf{on}}$. In the first phase of every round, we locally improve the strategy $C_{\mathcal{P}}^{\mathsf{off},\phi}$ computed in Section IV-C that aims to visit an accepting state of the chosen ASCC $\mathcal{U}$. In each step of the resulting strategy $C_{\mathcal{P}}^{\mathsf{on},\phi}$, we consider the set $\mathrm{Run}_{\phi}(u_{i,k})$ of all finite runs that start in the current state $u_{i,k}$ and lead to an accepting state from the set $F_{\mathcal{U}}$ with all transitions shortening in the indicator $I_{F_{\mathcal{U}}}$ defined according to (7), i.e.,

$$\begin{aligned} \mathrm{Run}_{\phi}(u_{i,k}) = \big\{\sigma \in \mathrm{Run}_{\mathrm{fin}}^{\mathcal{P}}(u_{i,k}) \mid last(\sigma) \in F_{\mathcal{U}}, \text{ and} \\ \forall 0 \le j \le |\sigma| - 1 : I_{F_{\mathcal{U}}}\left((\sigma(j), \sigma(j+1))\right) = 1\big\}. \end{aligned}$$

Let $\sigma \in \mathrm{Run}_{\phi}(u_{i,k})$ be the run that minimizes the optimization function f from (6). Then $C_{\mathcal{P}}^{\mathsf{on},\phi}(\sigma_{\mathrm{all}}) = \sigma(1)$. Just like in the offline algorithm, the strategy $C_{\mathcal{P}}^{\mathsf{on},\phi}$ is applied until a state from the set $F_{\mathcal{U}}$ is visited. In the second phase of strategy $C_{\mathcal{P}}^{\mathsf{on}}$, we locally improve the strategy $C_{\mathcal{P}}^{\mathsf{off},V}$ for the average subgoal computed in Section IV-C and we use $C_{\mathcal{P}}^{\mathsf{on},V}$ to denote the resulting strategy. At the beginning of the second phase of the current round i, we aim to reach the cycle $\mathsf{cyc}_{\mathcal{U}}^V = c_0 \ldots c_m$ of the ASCC $\mathcal{U}$ and we use the same idea that is used in the first phase above. To be specific, we define $C_{\mathcal{P}}^{\mathsf{on},V}(\sigma_{\mathrm{all}}) = \sigma(1)$, where σ is the run minimizing f from the set

$$\begin{aligned} \mathrm{Run}_V^{\mathsf{init}}(u_{i,k}) = \Big\{\sigma \in \mathrm{Run}_{\mathrm{fin}}^{\mathcal{P}}(u_{i,k}) \mid last(\sigma) \in \mathsf{cyc}_{\mathcal{U}}^V, \text{ and} \\ \forall 0 \le j \le |\sigma| - 1 : I_{\mathsf{cyc}_{\mathcal{U}}^V}\left((\sigma(j), \sigma(j+1))\right) = 1\Big\}. \end{aligned}$$

Once a state c_aß$\mathsf{cyc}_{\mathcal{U}}^V$ of the cycle is reached, we continue as follows. Let $c_b \in \mathsf{cyc}_{\mathcal{U}}^V$ be the first state labeled with π_{sur} that is visited from c_a if we follow the cycle. Until we reach the state c_b, the optimal finite run σ is chosen from the set

$$\begin{aligned} \mathrm{Run}_V(u_{i,k}) = \big\{\sigma \in \mathrm{Run}_{\mathrm{fin}}^{\mathcal{P}}(u_{i,k}) \mid last(\sigma) = c_b, \text{ and} \\ \forall 0 \le j \le |\sigma| - 1 : I_{c_b}((\sigma(j), \sigma(j+1))) = 1 \text{ or} \\ \left|\sigma_{c_a \to u_{i,k}}\right| + |\sigma| \le b - a + 2 \bmod (m+1)\big\} \end{aligned}$$

where $\sigma_{c_a \to u_{i,k}}$ is the finite run already executed in $\mathcal{P}$ from state c_a to the current state $u_{i,k}$. Intuitively, the set contains every finite run starting from the current state and leading to c_b that either has all transitions shortening in I_{c_b} or the length of the finite run is such that if we were to perform the finite run, the length of the performed run from c_a to c_b would not be longer than following the cycle from c_a to c_b. When state c_b is reached, we restart the above procedure with $c_a = c_b$. The strategy $C_{\mathcal{P}}^{\mathsf{on},V}$ is performed until $l_i = \max\{j(1/i), i \cdot (k_i + |S_{\mathcal{U}}|) \cdot g_{\max}\}$ surveillance cycles are completed in the second phase of the current round i, where k_i is the number of steps of the first phase and j is from Prop. 1 (the value is given by the strategy $C_{\mathcal{P}}^{\mathsf{off}}$).

Theorem 2: The online control strategy C that results from projecting strategy $C^{\mathsf{on}}_{\mathcal{P}}$ from $\mathcal{P}$ to $\mathcal{T}$ guarantees satisfaction of the LTL specification ϕ. Moreover, it holds that the online control strategy provides lower or equal expected average cumulative penalty per surveillance cycle than the offline control strategy, i.e., $V_{\mathcal{T},C}(s_{\mathrm{init}}) = V_{\mathcal{P},C^{\mathsf{on}}_{\mathcal{P}}}(s_{\mathcal{P}init}) \leq V^*_{\mathcal{U}}$.

Proof: To prove that the online control strategy satisfies the LTL specification, we show that under every penalty profile Δ, the run $\rho_{C^{\mathsf{on}}_{\mathcal{P}},\Delta}(s_{\mathcal{P}init})$ induced by strategy $C^{\mathsf{on}}_{\mathcal{P}}$ visits the set $F_{\mathcal{P}}$ of accepting states infinitely many times. From the definitions of sets $\mathrm{Run}_{\phi}(u_{i,k})$, $\mathrm{Run}^{\mathsf{init}}_{V}(u_{i,k})$ and $\mathrm{Run}_{V}(u_{i,k})$ it follows that every round of $C^{\mathsf{on}}_{\mathcal{P}}$ ends after a finite number of steps and at least one accepting state is visited in every round. The inequality $V_{\mathcal{P},C^{\mathsf{on}}_{\mathcal{P}}}(s_{\mathcal{P}init}) \leq V^*_{\mathcal{U}}$ follows directly from the design of the strategy $C^{\mathsf{on}}_{\mathcal{P}}$. □

As we show in Section V, computation of the APPC value of the online strategy according to (2) is intractable even for reasonably small examples. Nevertheless, we demonstrate that in some cases, it can be computed based on the specifics of the transition system, and in general, it can be estimated from statistical results. The case studies also demonstrate that even though the construction of the online strategy does not guarantee strict decrease in APPC value in theory, these strategies often result in a significant improvement.

Complexity: To design the online strategy $C^{\mathsf{on}}_{\mathcal{P}}$, we first compute the strategy $C^{\mathsf{off}}_{\mathcal{P}}$, see Section IV-C. The complexity of one step of the online strategy is as follows. The cardinality of the set of finite runs $\mathrm{Run}_{\phi}(u_{i,k})$ grows exponentially with the minimum weight $w^*_{\mathcal{P}}(u_{i,k}, F_{\mathcal{U}})$. Analogously, the same holds for sets $\mathrm{Run}^{\mathsf{init}}_{V}(u_{i,k})$ and $\mathrm{Run}_{V}(u_{i,k})$, and the set cyc^*_V or one of its surveillance states. In the following discussion on the usability of the online control, we propose a simple rule to simplify the computations and effectively use the algorithm in real time. Also, the complexity of one step of the strategy $C^{\mathsf{on}}_{\mathcal{P}}$ grows exponentially with the user-defined planning horizon h and the system-specific visibility range v. Hence, h should be chosen wisely. One should also keep in mind that generally, the higher the planning horizon, the better local improvement.

Usability: Just like the offline strategy, the online strategy is not finite-memory in general. To reduce memory usage, we construct new strategy $\overline{C^{\mathsf{on}}_{\mathcal{P}}}$ from $C^{\mathsf{on}}_{\mathcal{P}}$ by applying the rule to reduce the number of performed surveillance cycles in the second phase of every round that was described in Section IV-C. Moreover, to simplify the computation in one step of the online control, we allow the user to define parameter $W_{\max} \geq \max\{w_{\mathcal{P}}((u,u'))|(u,u') \in T_{\mathcal{P}}\}$ that serves as follows. In the first phase of round i and at the beginning of the second phase, when not yet on the optimal APPC cycle, we only consider prefixes of the finite runs from the sets $\mathrm{Run}_{\phi}(u_{i,k})$, $\mathrm{Run}^{\mathsf{init}}_{V}(u_{i,k})$ or $\mathrm{Run}_{V}(u_{i,k})$, and the set cyc^*_V of weight at most $W_{\max}$. In the second phase of a round, when the optimal cycle has already been reached, if the weight of the fragment of the cycle from c_a to c_b is more than $W_{\max}$, we first optimize the run from c_a to an intermediate state c'_b for which it holds that the weight of the fragment of the cycle from c_a to c'_b is at most $W_{\max}$ but highest possible. Finally, we postpone the computation of the value $j(1/i)$ for round i for as long as possible in the same manner as for the offline strategy in Section IV-C. The strategy $\overline{C^{\mathsf{on}}_{\mathcal{P}}}$ is equivalent to $C^{\mathsf{on}}_{\mathcal{P}}$ in the meaning that it provably satisfies the LTL formula ϕ and has the same APPC value as the strategy $C^{\mathsf{on}}_{\mathcal{P}}$. The improvement of the usability of the online control is demonstrated in Section V.

Example 3: The online control for the delivery system from Ex. 1 that locally improves the offline control described in Ex. 2 works as follows. Consider planning horizon $h = 9$. In every step of the first phase of every round, the robot considers all finite runs that start in the current state, continue to the base location and end in the delivery location that should be visited next. The robot performs the first transition of the run minimizing the function from (6). When computing the value for a finite run, the penalties in states that are within the visibility range and would be visited within nine time units are simulated. In the second phase, the robot locally improves the finite run leading from one delivery location to the other, while visiting at most as many states as there are on the cycle shown in Fig. 4 between the two delivery locations. The number of surveillance cycles in the second phase of round i is $l_i = \max\{i \cdot (k_i + 576), j(1/i)\}$, where k_i is the number of steps in the first phase of the round. Unlike in the offline control in Ex. 2, number k_i might differ from round to round.

Note that the set of finite runs considered in one step of the online control can be very large, especially in the first phases of rounds when the finite runs are considerably long. We can use the parameter $W_{\max}$ introduced in the discussion on the usability of the online control above to decrease the number and weight of finite runs considered in every step of the control. By applying the rest of the rules from the discussion, we can also decrease the number of visits to delivery locations in every round and thus visit the base location more often.

Remark 4: The online control introduced in this section is in fact a heuristic. We can formulate other heuristics that would construct a strategy satisfying Th. 2. For example, consider a strategy that is constructed from $C^{\mathsf{off}}_{\mathcal{P}}$ from Section IV-C in the same way as the strategy $C^{\mathsf{on}}_{\mathcal{P}}$, i.e., deploying sets $\mathrm{Run}_{\phi}(u_{i,k})$ and $\mathrm{Run}^{\mathsf{init}}_{V}(u_{i,k})$, except in the second phase of every round, once a state $c_a \in \mathsf{cyc}^V_{\mathcal{U}}$ on the cycle is reached, the optimal finite run is chosen from the set defined as

$$\begin{aligned}\mathrm{Run}_V(u_{i,k}) = \big\{\sigma \in \mathrm{Run}^{\mathcal{P}}_{\mathrm{fin}}(u_{i,k}) \mid \mathit{last}(\sigma) = c_b, \text{ and} \\ \forall 0 \leq j \leq |\sigma| - 1 : I_{c_b}\left((\sigma(j), \sigma(j+1))\right) = 1 \text{ or} \\ \left|\sigma_{c_a \to u_{i,k}}\right| + |\sigma| \leq 2 \cdot (b - a + 2 \bmod (m+1))\big\}\end{aligned}$$

i.e., we consider all finite runs consisting only of transitions shortening in I_{c_b} and all finite runs leading to the target state c_b with length at most twice the length of following the cycle from c_a to c_b. We refer to the projection of this strategy from $\mathcal{P}$ to $\mathcal{T}$ as the *modified online control*.

We can define other heuristics in a similar way by changing only the definition of the sets of finite runs $\mathrm{Run}_{\phi}(u_{i,k}), \mathrm{Run}_{V}(u_{i,k})$. However in order to guarantee satisfaction of Th. 2, the sets must satisfy the following conditions. The definition of $\mathrm{Run}_{\phi}(u_{i,k})$ guarantees that an accepting state from $F_{\mathcal{U}}$ is always visited after a finite number of steps. The definition of $\mathrm{Run}_{V}(u_{i,k})$ guarantees a visit of the cycle $\mathsf{cyc}^V_{\mathcal{U}}$ after at most $|S_{\mathcal{U}}|$ steps and once a state $c_a \in \mathsf{cyc}^V_{\mathcal{U}}$ on the cycle is reached, the set $\mathrm{Run}_{V}(u_{i,k})$ guarantees visit of the state c_b in a finite number of steps.

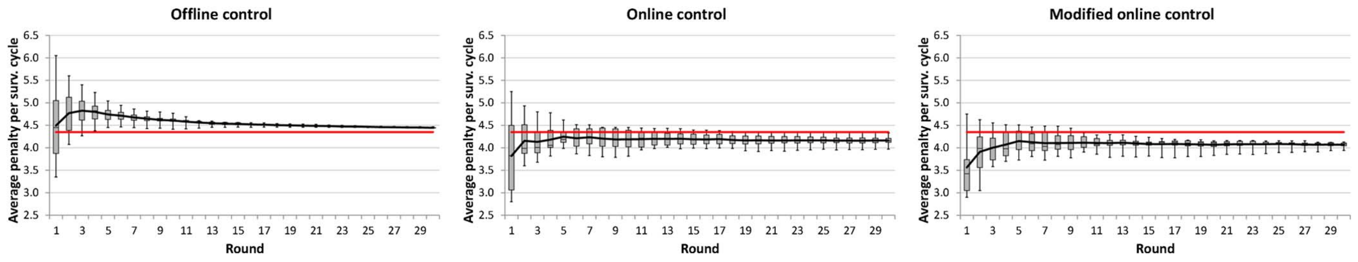


Fig. 5. Evolution of the average penalty per surveillance cycle obtained in simulations of the offline, online and modified online control for the delivery system case study. For each control, the statistics is built on 10 individual runs of 30 rounds each. The red line marks the optimal APPC value and the black line shows the mean over executed runs.

V. Implementation and Case Studies

A. Implementation

The framework presented in this paper is implemented in our simulation tool ConTool [32]. Input transition system can be defined in DOT language and then visualized using Graphviz [33]. The MCs defining the penalties in states are loaded from a text file. We use LTL2BA [25] to generate a Büchi automaton for given LTL formula. The user can choose to simulate the offline, online or modified online control from Rem. 4. All three control strategies implement the rules introduced in Sections IV-C and D to reduce memory usage and computational costs. After specifying additional parameters such as visibility range, planning horizon and the parameter $W_{\max}$ from Section IV-D, the simulation tool allows to observe the control one transition at a time.

B. Case Study 1: Delivery System

The offline and online control strategies for the delivery system from Ex. 1 were described in Ex. 2 and 3, respectively. Here we report on the results we obtained from executing 10 runs of 30 rounds for all three types of control strategies using our implementation from Section V-A. In simulations of the online and modified online controls, we used parameter $W_{\max} = 9$. We present the analysis of the average penalty per surveillance cycle at the end of every round in Fig. 5. For the offline control, the value gradually converges to the (optimal) APPC value 4.35 of the offline control, marked as a red line in Fig. 5. On the other hand, for both online and modified online control strategies, the average is below 4.35 due to the local improvement based on local sensing. Due to the size and density of the transition system, it is intractable to compute the exact APPC value as defined in (2) for the online and modified online control. Nevertheless, from Fig. 5 we can observe that the average penalty per surveillance cycle stabilizes in timely manner at approximately 4.16 for the online control and 4.05 for the modified online control.

The number of surveillance cycles performed in the second phase of every round i using the rules from Sections IV-C and D was always less than $i \cdot (k_i + 568)$ for all three types of control, i.e., the second phase always ended due to the fact that the average incurred in the round was below the threshold $V^*_{\mathcal{U}} + (2/i)$. That means, we were never forced to compute the value $j(1/i)$. The maximum number of surveillance cycles performed in the second phase of a round of offline control strategy was 636, average was only 29 and median was 8. Using online control strategy, the maximum number of surveillance cycles performed in the second phase of a round was 10, the average was 2 and the median was 1. Similarly for the modified online strategy, the maximum was 9 and both average and median were 1, i.e., for all three control strategies the rules from Sections IV-C and IV-D reduced the number of performed surveillance cycles in every round substantially from thousands to only tens or few hundreds and thus allowed to visit the base location much more often. Moreover, the number of surveillance cycles in the second phase of a round did not evolve monotonically, rather randomly.

We ran the simulations on a Lenovo laptop with Windows 7, Intel Pentium CPU 2.00 GHz and 3 GB RAM. The offline strategy was computed in 45 seconds on average. One step of the online and modified online control took 150 milliseconds and 7.5 seconds on average, with 100 milliseconds and 12 seconds deviation, respectively.

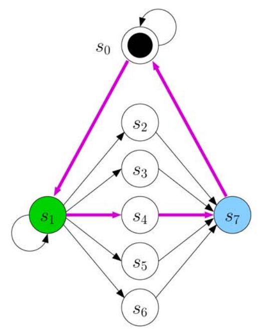
C. Case Study 2: Stock Market

The second case study we use to evaluate our framework models a simple stock market and a broker that performs one action on the market at a time. He can sell or buy stocks, or decide to wait. We assume that the system can be modeled as the transition system depicted in Fig. 6(a). The system starts in state s_0. The broker decides his next action in state s_1 and he can choose from five different buying and selling orders. All transitions have weight 1. In Fig. 6(b), we list the transition matrices of the Markov chains that define penalties in states of the TS. The initial distribution is always the uniform distribution over the possible values of the penalty in a given state. Only the states $s_2, \ldots, s_6$ can have nonzero penalty modeling the fact that only buying and selling stocks has any value. Finally, state s_7, in blue, can be seen as an evaluation state, where the gains and losses are counted. The visibility range v is 4, i.e., the broker can always observe penalties in all states.

The mission for the broker is to make an infinite number of orders and, at the same time, to minimize the penalty incurred per order. We model this requirement with LTL formula

$$\mathbf{GF}\, a \;\wedge\; \mathbf{GF}\, \pi_{\text{sur}}$$

where a and π_{sur} are true in state s_7. The Büchi automaton generated for the formula using [25] has four states, the product has one ASCC with 25 states. The optimal APPC cycle projected to



(a)

$$A_0 = A_1 = A_7 = \begin{array}{c c} & 0 \\ 0 & \left(\begin{array}{c} 1.0 \end{array}\right) \end{array} \qquad A_2 = \begin{array}{c c} & 2 \\ 2 & \left(\begin{array}{c} 1.0 \end{array}\right) \end{array} \qquad A_3 = \begin{array}{c c} & \begin{array}{cc} 1 & 2 \end{array} \\ \begin{array}{c} 1 \\ 2 \end{array} & \left(\begin{array}{cc} 0.8 & 0.2 \\ 0.2 & 0.8 \end{array}\right) \end{array}$$

$$A_4 = \begin{array}{c c} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \left(\begin{array}{cccc} 0.1 & 0.5 & 0.3 & 0.1 \\ 0.1 & 0.5 & 0.3 & 0.1 \\ 0.1 & 0.5 & 0.3 & 0.1 \\ 0.1 & 0.5 & 0.3 & 0.1 \end{array}\right) \end{array} \quad A_5 = \begin{array}{c c} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \left(\begin{array}{cccc} 0.2 & 0.8 & 0 & 0 \\ 0 & 0.2 & 0.8 & 0 \\ 0 & 0 & 0.2 & 0.8 \\ 0.8 & 0 & 0 & 0.2 \end{array}\right) \end{array} \quad A_6 = \begin{array}{c c} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \left(\begin{array}{cccc} 0 & 1.0 & 0 & 0 \\ 0 & 0 & 1.0 & 0 \\ 0 & 0 & 0 & 1.0 \\ 0.9 & 0 & 0 & 0.1 \end{array}\right) \end{array}$$

(b)

Fig. 6. (a) Transition system that models a broker acting on a simple stock market. In state s_1, in green, the broker chooses one of the five buying or selling orders. All transitions have weight 1. The optimal cycle with respect to the expected average penalty per surveillance cycle is shown in magenta. (b) Matrices of Markov chains that define penalties in states of the transition system. Matrix A_i describes penalty in state s_i.

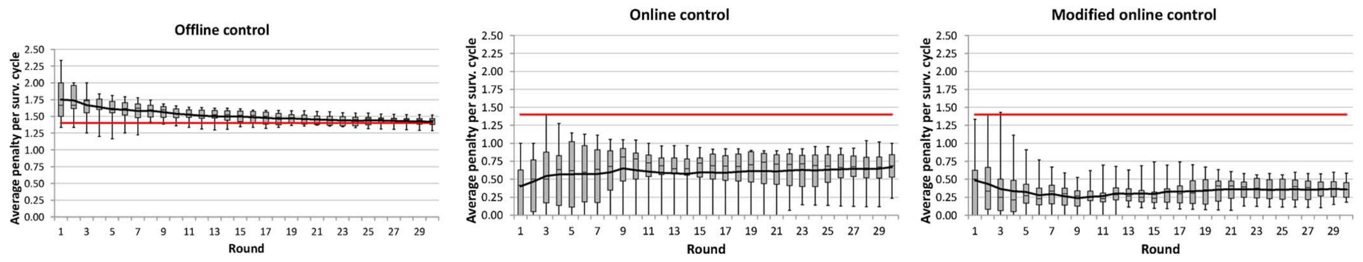


Fig. 7. Evolution of the average penalty per surveillance cycle attained in simulations of the offline, online and modified online control for the stock market case study. For each control strategy, the statistics is built on 10 individual runs of 30 rounds each. The red line marks the optimal APPC value and the black line shows the mean over executed runs.

the transition system is shown in magenta in Fig. 6(a) and the optimal APPC value is 1.4.

We present statistical results that we obtained by running 10 runs of 30 rounds for each of the offline, online and modified online control strategies. In all simulations we used planning horizon $h = 9$ and we did not use the parameter $W_{\max}$. In Fig. 7, we plot the average penalty per surveillance cycle at the end of every round. The red line marks the APPC value of the offline control 1.4. For the offline control, the obtained value converges to the optimal APPC value fairly fast. We can observe considerable improvement for both online and modified online control. The reason is the following. In the offline control, when in the second phase of a round the broker always chooses the action leading to state s_4 that has the minimum expected value g_E. On the other hand, in the same situation when using online control, the broker always chooses the next action according to the simulated expected values of penalties g_{sim} rather than their expected values g_E. Finally, using the modified online control, the broker is allowed to wait up to three time units in state s_1 and only then decide to buy or sell. Note that the penalty in state s_6 gradually increases from 0 to 3 and then with probability 90% it drops to 0 again. The broker waits in s_1 until the penalty in state s_6 has value 3 and then moves to s_6. Just like for the case study in Section V-B, it is intractable to compute the APPC value for the online and modified online control. However, from the discussion above, we can conclude that the APPC value of the modified online control is 0.3 and this fact can also be observed in Fig. 7. Based on Fig. 7, the APPC value of the online control strategy is approximately 0.64.

The number of surveillance cycles in the second phase of every round using the rules in Sections IV-C and D was always below $i \cdot (k_i + 25)$ for all three strategies, i.e., every round ended with the average penalty per surveillance cycle in the round dropping below the threshold $V_{\mathcal{U}}^* + (2/i)$ and we never needed to compute the value $j(1/i)$. The maximum number of surveillance cycles performed in the second phase of a round was 288 for the offline control, 12 for the online control and 5 for the modified online control, and the median was 1 in all three cases. Hence, the improvement of the rules in Sections IV-C and D is again remarkable. In all three cases, the number of surveillance cycles in rounds did not evolve monotonically, rather randomly.

We ran the simulations on a Lenovo laptop with Windows 7, Intel Pentium CPU 2.00 GHz and 3 GB RAM. The offline strategy took 0.5 seconds on average to compute. One step of the online and modified online control strategies always took under one millisecond.

VI. Conclusion and Future Work

In this paper, we considered the problem of synthesizing an optimal control strategy for a deterministic transition system under temporal logic constraints. We assumed real-valued penalties with probabilistic behaviors in the states of the system. We constructed a control strategy that, while guaranteeing satisfaction of an LTL formula, minimizes the expected average penalty per visit of a desired set of states. We also presented (a class of) control strategies that use local sensing of the penalties in real time and simulation of their values over finite horizon to improve the average penalty per visit of the set of states in every execution of the system.

The framework presented in this paper can be extended to Markov decision processes (MDPs). Our preliminary results on this topic can be found in [34], where we consider the analogous problem for MDPs with static costs. This problem was also investigated in [35]. The authors present a solution that is optimal only in special cases. In [34], we prove that our approach results in an optimal solution for any MDP and LTL formula. We also strongly believe that a similar approach can be used for nondeterministic transition systems but one first needs to properly define the optimization objective APPC that accounts for the nondeterminism of transitions.

REFERENCES

[1] C. Baier and J. Katoen, *Principles of Model Checking*. Cambridge, U.K.: MIT Press, 2008.
[2] A. Hinton, M. Kwiatkowska, G. Norman, and D. Parker, "PRISM: A tool for automatic verification of probabilistic systems," in *Proc. TACAS, ser. LNCS*, 2006, pp. 441–444.
[3] J. Barnat, L. Brim, V. Havel, J. Havlíček, J. Kriho, M. Lenčo, P. Ročkai, V. Štill, and J. Weiser, "DiVinE 3.0—An Explicit-State Model Checker for Multithreaded C & C++ Programs," in *Proc. CAV*, *ser. LNCS*, 2013, pp. 863–868.
[4] S. L. Smith, J. Tmová, C. Belta, and D. Rus, "Optimal path planning for surveillance with temporal-logic constraints," *Int. J. Rob. Res.*, vol. 30, no. 14, pp. 1695–1708, 2011.
[5] J.-F. Raskin, K. Chatterjee, L. Doyen, and T. A. Henzinger, "Algorithms for Omega-Regular Games with Imperfect Information," *Logical Meth. Comput. Sci.*, vol. 3, no. 3:4, pp. 1–23, 2007.
[6] M. Kwiatkowska and D. Parker, "Automated verification and strategy synthesis for probabilistic systems," in *Proc. ATVA*, *ser. LNCS*, 2013, pp. 5–22.
[7] P. Tabuada and G. Pappas, "Linear time logic control of discrete-time linear systems," *Trans. Autom. Control*, vol. 51, no. 12, pp. 1862–1877, Dec. 2006.
[8] E. A. Gol, M. Lazar, and C. Belta, "Language-guided controller synthesis for discrete-time linear systems," in *Proc. HSCC*, 2012, pp. 95–104.
[9] T. Wongpiromsarn, U. Topcu, and R. R. Murray, "Receding horizon temporal logic planning for dynamical systems," in *Proc. CDC*, 2009, pp. 5997–6004.
[10] M. Kloetzer and C. Belta, "A fully automated framework for control of linear systems from temporal logic specifications," *Trans. Autom. Control*, vol. 53, no. 1, pp. 287–297, Jan. 2008.
[11] B. Yordanov, J. Tumova, I. Cerna, J. Barnat, and C. Belta, "Temporal logic control of discrete-time piecewise affine systems," *Trans. Autom. Control*, vol. 57, no. 6, pp. 1491–1504, Jun. 2012.
[12] J. Ding, M. Kamgarpour, S. Summers, A. Abate, J. Lygeros, and C. Tomlin, "A stochastic games framework for verification and control of discrete time stochastic hybrid systems," *Automatica*, vol. 49, no. 9, pp. 2665–2674, 2013.
[13] Y. Tazaki and J. Imura, "Discrete abstractions of nonlinear systems based on error propagation analysis," *Trans. Autom. Control*, vol. 57, no. 3, pp. 550–564, Mar. 2012.
[14] G. Pola, A. Girard, and P. Tabuada, "Approximately bisimilar symbolic models for nonlinear control systems," *Automatica*, vol. 44, no. 10, pp. 2508–2516, 2008.
[15] G. Reissig, "Computing Abstractions of Nonlinear Systems," *Trans. Autom. Control*, vol. 56, no. 11, pp. 2583–2598, Nov. 2011.
[16] M. Zamani, G. Pola, M. Mazo, and P. Tabuada, "Symbolic models for nonlinear control systems without stability assumptions," *Trans. Autom. Control*, vol. 57, no. 7, pp. 1804–1809, Jul. 2012.
[17] C. Szepesvari, *Algorithms for Reinforcement Learning*. San Rafael, CA, USA: Morgan & Claypool, 2010.
[18] E. M. Wolff, U. Topcu, and R. M. Murray, "Optimal control with weighted average costs and temporal logic specifications," in *Proc. RSS*, 2012.
[19] X. C. Ding, M. Lazar, and C. Belta, "Receding horizon temporal logic control for finite deterministic systems," in *Proc. ACC*, 2012, pp. 715–720.
[20] M. Svoreňová, J. Tmová, J. Barnat, and I. Černá, "Attraction-based receding horizon path planning with temporal logic constraints," in *Proc. CDC*, 2012, pp. 6749–6754.
[21] M. Svoreňová, I. Černá, and C. Belta, "Optimal receding horizon control for finite deterministic systems with temporal logic constraints," in *Proc. ACC*, 2013, pp. 4399–4404.
[22] M. Y. Vardi and P. Wolper, "Reasoning about infinite computations," *Inf. Comput.*, vol. 115, no. 1, pp. 1–37, 1994.
[23] P. Gastin and D. Oddoux, "Fast LTL to Büchi automata translation," in *Proc. CAV*, 2001, pp. 53–65.
[24] F. Somenzi and R. Bloem, "Efficient Büchi automata from LTL formulae," in *Proc. CAV*, 2000, pp. 248–263.
[25] P. Gastin and D. Oddoux, LTL 2 BA: Fast translation from LTL formulae to Büchi automata, 2001. [Online]. Available: http://www.lsv.ens-cachan.fr/~gastin/ltl2ba/
[26] J. R. Norris, *Markov Chains*. Cambridge, U.K.: Cambridge Univ. Press, 1998.
[27] D. Bertsekas, *Dynamic Programming and Optimal Control, vol. II*. Nashua, NH, USA: Athena Scientific, 2007.
[28] K. Chatterjee and L. Doyen, "Energy and mean-payoff parity Markov decision processes," in *Proc. MFCS*, 2011, pp. 206–218.
[29] R. Ash and C. Doléans-Dade, *Probability & Measure Theory*. New York, NY, USA: Academic, 2000.
[30] R. M. Karp, "A characterization of the minimum cycle mean in a digraph," *Discrete Math.*, vol. 23, no. 3, pp. 309–311, 1978.
[31] R. Tarjan, "Depth-first search and linear graph algorithms," in *Proc. SWAT*, 1971, pp. 114–121.
[32] M. Svoreňová, L. Mařica, and I. Černá, ConTool—Tool for simulation of control algorithms, 2013. [Online]. Available: http://www.fi.muni.cz/~x175388/contool.html
[33] E. R. Gansner and S. C. North, "An open graph visualization system and its applications to software engineering," *Software—Practice Exper.*, vol. 30, no. 11, pp. 1203–1233, 2000.
[34] M. Svoreňová, I. Černá, and C. Belta, "Optimal control of MDPs with temporal logic constraints," in *Proc. CDC*, 2013, pp. 3938–3943.
[35] X. C. Ding, S. Smith, C. Belta, and D. Rus, "MDP optimal control under temporal logic constraints," in *Proc. CDC*, 2011, pp. 532–538.

Mária Svoreňová (S'14) received the B.S. and M.Sc. degrees in discrete mathematics and algebra and the B.S. degree in computer science from Masaryk University, Brno, Czech Republic, in 2008, 2011, and 2010, respectively. She is currently pursuing the Ph.D. degree in computer science at Masaryk University.

Her research interests include formal methods, temporal logics, model checking, game theory, and controller synthesis.

Ivana Černá received the M.Sc. and Ph.D. degrees in computer science from Comenius University, Bratislava, Slovak Republic, in 1986 and 1992, respectively.

She is a Professor at Faculty of Informatics, Masaryk University, Brno, Czech Republic. Her research interests include theory of communicating and parallel systems, formal verification and verification tools, algorithm design, and analysis. She is a coauthor of algorithms implemented in a parallel and distributed verification tool DiVinE.

Calin Belta (SM'11) received M.Sc and Ph.D. degrees in mechanical engineering from the University of Pennsylvania.

He is an Associate Professor in the Department of Mechanical Engineering, Department of Electrical and Computer Engineering, and the Division of Systems Engineering, Boston University, Boston, MA, USA. He is an Associate Editor for the *SIAM Journal on Control and Optimization (SICON)*. His research focuses on dynamics and control theory, with particular emphasis on hybrid and cyber-physical systems, formal synthesis and verification, and applications in robotics and systems biology.

Dr. Belta received the Air Force Office of Scientific Research Young Investigator Award and the National Science Foundation CAREER Award.