Controlled Markov Processes With Safety State Constraints

Mahmoud El Chamie, Yue Yu, Behçet Açıkmeşe, and Masahiro Ono

***Abstract*—This paper considers a Markov decision process (MDP) model with safety state constraints, which specify polytopic invariance constraints on the state probability distribution (*pd*) for all time epochs. Typically, in the MDP framework, safety is addressed indirectly by penalizing failure states through the reward function. However, such an approach does not allow imposing hard constraints on the state *pd*, which could be an issue for practical applications where the chance of failure must be limited to prescribed bounds. In this paper, we explicitly separate state constraints from the reward function. We provide analysis and synthesis methods to impose generalized safety constraints at all time epochs, unlike current constrained MDP approaches where such constraints can only be imposed on the stationary distributions. We show that, contrary to the unconstrained MDP policies, optimal safe MDP policies depend on the initial state *pd*. We present novel algorithms for both finite- and infinite-horizon MDPs to synthesize feasible decision-making policies that satisfy safety constraints for all time epochs and ensure that the performance is above a computable lower bound. Linear programming implementations of the proposed algorithms are developed, which are formulated by using the duality theory of convex optimization. A swarm control simulation example is also provided to demonstrate the use of proposed algorithms.**

***Index Terms*—Markov processes, dynamic programming, agents and autonomous systems, stochastic optimal control, constrained control, Markov decision processes, controlled Markov chains.**

I. Introduction

MARKOV decision processes (MDPs) have been successfully utilized to model various decision-making problems [1]–[3]. With the growing interest in cyber-physical systems (CPS), there is an increasing need to incorporate state constraints into MDP models to capture physical limitations and mission safety requirements. Having the ability to integrate such constraints also facilitates the rigorous construction of hierarchical decision-making architectures since it enables a high-level MDP-based planner to explicitly consider the capabilities and limitations of a low-level planner (e.g., a physical control system). Example applications include autonomous vehicles with limited speed/acceleration, communication range, and operational space, and multivehicle systems performing missions in hazardous regions, with collision/conflict avoidance [4]–[7].

Typically, in MDPs, safety is addressed by imposing penalty on undesirable states. This paper proposes to separate safety state constraints from the reward function by explicitly including these constraints in the MDP formulation. In particular, this paper considers a general class of safety state constraints in the MDP model, which imposes linear polytopic invariance constraints on the state probability distribution (*pd*) for all time epochs. An example of a safety constraint would be to limit the frequency of visiting a potentially hazardous state by an agent. Another example is given by multiagent systems where the states have capacity limits, i.e., the expected number of agents in each state must remain below a prescribed upper bound [5]. Hence, this general class of safety also includes safety upper bound constraints studied in Markov chain (MC) literature [8]–[10]. Since the time evolution of the state *pd* vector for an MDP is determined by the underlying MC emerging from the decision policy, we can view the problem as an MDP policy synthesis problem to obtain a safe MC. An important feature of safe decision policies is that, in many cases, they must be randomized policies rather than deterministic ones. Existence of safety state constraints on the underlying MC can eliminate the feasibility of deterministic policies, which are optimal for the standard MDPs without such constraints [11].

The classical approach to policy synthesis in the presence of constraints is based on state-action frequencies [12]. This approach provides asymptotic satisfaction of safety constraints for stationary distributions; however, it does not necessarily satisfy the hard constraints on the transient regime as the ones imposed by the safety upper bound constraints of this paper. Furthermore, this paper extends the type of safety state constraints to a larger class. By considering MDPs with finite number of states and actions subject to hard safety constraints on the state *pd*, we devise efficient linear-programming-based (LP-based) policy synthesis algorithms, which are obtained by using the duality theory of convex optimization. The idea behind the devised algorithms

Manuscript received March 3, 2017; revised October 23, 2017 and March 7, 2018; accepted April 10, 2018. Date of publication June 21, 2018; date of current version February 26, 2019. This research was supported in part by the Defense Advanced Research Projects Agency under Grant D14AP00084, in part by the National Science Foundation under Grant CNS-1624328, and in part by the Office of Naval Research under Grant N00014-15-IP-00052. Recommended by Associate Editor Sean B. Andersson. *(Corresponding author: Mahmoud El Chamie.)*

M. El Chamie is with the Systems Department, United Technologies Research Center, East Hartford, CT 06108 USA (e-mail: elchamm@utrc.utc.com).

Y. Yu and B. Açıkmeşe are with the Department of Aeronautics and Astronautics, University of Washington, Seattle, WA 98195 USA (e-mail: yueyu@uw.edu; behcet@uw.edu).

M. Ono is with NASA's Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109 USA (e-mail: masahiro.ono@jpl.nasa.gov).

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.2018.2849556

is to optimize the reward over a convex set of all feasible safe policies while guaranteeing that the expected total reward is above a computable lower bound. The contribution of the paper is threefold.

1) To the best of our knowledge, this paper is the first to separate safety state constraints from the reward function by explicitly incorporating these constraints in the problem formulation of finite- and infinite-horizon MDPs. Thus, the model is capable of imposing polytopic constraints on the state *pd* for both the transient and the stationary regimes, i.e., for all time epochs.
2) We show that, contrary to the unconstrained MDP policies, optimal *safe* MDP policies depend on the initial state *pd*. We, accordingly, devise new policy synthesis algorithms based on dynamic programming (DP) for different levels of uncertainties on the initial distribution: worst-case policy synthesis for unknown *pd*, robust policy synthesis for bounded *pd*, and forward projection policy synthesis for a known *pd*.
3) Using the duality theory, we provide an LP-based implementation of the proposed algorithms, which allows efficient and reliable computation of the decision policies by using standard LP solvers.

While the proposed algorithms are applicable to any MDP system with safety state constraints, we present an autonomous multiagent system example to demonstrate the algorithms. The rest of the paper is organized as follows. Section II lists the related work on MDPs and compares this paper's contribution to the existing work in the constrained MDP literature. Section III provides the notation used in the paper and gives some MDP preliminaries. Section IV introduces the general safety constrained MDP (SC-MDP) model studied in this paper. Section V shows that the classical constrained MDP solutions can violate the safety state constraints introduced in this paper. Section VI provides the DP solution approach for solving SC-MDP problems, and also presents the algorithms for their policy synthesis. Simulations for a swarm coordination problem are presented in Section VII to demonstrate the proposed solution algorithms. Section VIII concludes the paper.

II. Related Research

Constraints in MDPs have been used to handle multiple design objectives where decisions are computed to maximize rewards for one objective, while guaranteeing the value of the other objective is within a desired range [12]. The constraints can also be imposed by the underlying operational environment, e.g., safety constraints in multiagent autonomous systems [4], [13], [14], or constraints in telecommunication applications [12]. When a system is required to avoid undesirable states, a typical approach is to include penalty in the reward function, with the idea that the resulting policy will keep the chance of visiting the undesirable states low. However, the reward-based penalty approach has several drawbacks. First, it is not clear how to choose the value of penalties, especially for MDPs with a large number of states. Second, it can only guarantee constraints for the stationary distribution, giving no guarantees on the transient behavior of the system.

The previous research has focused on finding optimal infinite-horizon stationary policies for constrained MDPs. Due to the constraints, the optimal policies might no longer be deterministic and stationary [15]. Reference [16] gave an example of a transient multichain MDP with state constraints and showed that the optimal policy is not necessarily stationary. For stationary policies to be optimal, specific assumptions on the underlying MC (such as regularity) are often used [17]. Optimal stationary policies for these specific models can be obtained by using algorithms based on LP [12].

The results of [9] and [18] on a constrained MDP model are closely related to some of our results. However, in addition to the difference that their approach is for infinite-horizon average reward MDPs as compared to the finite-horizon total discounted reward in this paper, there is also a fundamental difference in the problem formulation and the definition of safety. While their approach seeks for safe initial conditions for a given MC, we aim to synthesize control policies that ensure safety for any safe initial *pd*.

Constrained MDPs are also related to the literature on formal verification methods [19]–[22], e.g., probabilistic model checking [23], [24, Ch. 10], and the corresponding formal verification tools such as PRISM [25]. Formal verification tools usually assume time-homogeneous MDPs, while the problem formulation in this paper relaxes this assumption by considering time-inhomogeneous MDPs. Furthermore, safety constraints considered in this paper are invariance constraints on the state *pd*, which is different from the safety constraints considered in formal methods framework.

More recently, MDPs with constraints are applied in path planning for robotics [26], flight control problems [27], and dynamic traffic assignment of artificial intelligence [5]. State constraints also allow direct comparisons with the chance-constrained optimal control [28], [29] and constrained norm minimization problems [30].

III. Preliminaries and Notation

For completeness, we define in this section the notation used in the paper, which is fairly standard in MC and MDP literature [11]. We will formulate the problem as a finite-horizon MDP and provide corresponding policy synthesis algorithms. We will then extend these algorithms to infinite horizon.

A. States and Actions

Let $\mathcal{S} = \{1, \ldots, n\}$ be a finite set of states and $\mathcal{A} = \{1, \ldots, p\}$ be a finite set of actions. Let X_t and A_t be random variables corresponding to the state and action at the tth time epoch. $\mathbf{e}_s$ is a vector of all zeros except for the sth element, which is equal to 1, $\mathbf{1}$ is the vector of all ones, and $\odot$ denotes the elementwise, Hadamard, product.

B. Decision Rule and Policy

We define a *decision rule* P_t at temporal epoch t to be the following function $P_t : \mathcal{S} \times \mathcal{A} \to [0, 1]$ that defines, for every state $s \in \mathcal{S}$, a *pd* over the set of actions $\mathcal{P}(\mathcal{A})$ as follows. For any state $s \in \mathcal{S}$ and any action $a \in \mathcal{A}$, we have

$P_t(s,a) := \text{Prob}[A_t = a | X_t = s]$. In typical MDP problems, the decision rule is a decision variable designed to optimize a given performance metric. With a little abuse of notation, the rule can be represented by a *decision matrix* $P_t \in \mathbb{R}^{n,p}$, where the entry of the sth row and ath column is the decision variable $P_t(s,a)$. Let

$$\pi = (P_0, P_1, \ldots, P_{N-1})$$

be the decision-making *policy* for N decision epochs. Let Π be the set of all possible policies defined as follows:

$$\Pi := \{\pi : P_t \mathbf{1} = \mathbf{1}, P_t \geq 0, t = 0, \ldots, N-1\}$$

where $\geq$ is elementwise inequality. Note that this decision rule has a Markovian property because the decision depends only on the current state without consideration to the history that led to this state. For unconstrained MDP problems, it is shown that there exists an optimal policy that is deterministic and Markovian [11]. This paper considers only Markovian policies, and the study of history-dependent policies is a subject of future research.

C. Rewards

Given a state $s \in \mathcal{S}$ and action $a \in \mathcal{A}$, we define the reward $R_t(s,a) \in \mathbb{R}$ for $t = 0, \ldots, N-1$ to be a real number, let $R_t \in \mathbb{R}^{n,p}$ be the matrix having the elements $R_t(s,a)$. The last stage of MDP when $t = N$ does not have an action decision, and thus the rewards can be given by a vector $\bar{\mathbf{r}}_N \in \mathbb{R}^n$ that gives a reward for every possible terminal state.

D. State Transitions

We define the MDP transition probability at time epoch t from a state i to a state j given an action k to be

$$G_{t,k}(i,j) := \text{Prob}[X_{t+1} = j | X_t = i, A_t = k].$$

Let $G_{t,k} \in \mathbb{R}^{n,n}$ be a row stochastic matrix having elements $G_{t,k}(i,j)$, and let $\mathcal{G}$ be the set having all transition probabilities.

For a given policy π, the elements of the induced MC transition matrix $M_t \in \mathbb{R}^{n,n}$ are

$$\begin{aligned} M_t(i,j) &:= \text{Prob}[X_{t+1} = j | X_t = i] \\ &= \sum_{k \in \mathcal{A}} P_t(i,k) G_{t,k}(i,j). \end{aligned} \tag{1}$$

To emphasize the explicit dependence of the MC transition matrix M_t on the decision matrix P_t, we will sometimes denote the transition matrix by M_{P_t}. Let $\mathbf{p}_t(i) = \text{Prob}[X_t = i]$ be the probability of being at state i at time t, and $\mathbf{p}_t \in \mathbb{R}^n$ be the vector of these probabilities.[1] The probability vector $\mathbf{p}_t$ is the state *pd* whose dynamics are given by the following lemma.

Lemma 1: The state *pd* vector $\mathbf{p}_t$ evolves according to the following equation:

$$\mathbf{p}_{t+1} = M_t^T \, \mathbf{p}_t, \quad t = 0, \ldots, N-1 \tag{2}$$

where M_t is a linear function of the decision matrix P_t

$$M_t = M_{P_t} = \sum_{k=1}^{p} G_{t,k} \odot \left((P_t \mathbf{e}_k) \mathbf{1}^T \right). \tag{3}$$

[1]In a multiagent setting, $\mathbf{p}_t$ is referred to as the density distribution of agents.

Proof: The proof follows directly from the definition of $\mathbf{p}_t(i)$ and $M_t(j,i)$, for all $i \in \mathcal{S}$, we have

$$\begin{aligned} \mathbf{p}_{t+1}(i) &= \text{Prob}[X_{t+1} = i] \\ &= \sum_{j \in \mathcal{S}} \text{Prob}[X_{t+1} = i | X_t = j] \text{Prob}[X_t = j] \\ &= \sum_{j \in \mathcal{S}} M_t(j,i) \mathbf{p}_t(j) \end{aligned}$$

and the lemma follows by using (1). ■

Remark 1: For a given policy π, the process $\{X_t, t = 1, 2, \ldots\}$ is a discrete-time MC whose state space is $\mathcal{S}$ and the transition probability matrix M_t is defined by the policy π. The discrete-time MC dynamical system is referred to as the *one-dimensional (1-D) stochastic system* in the rest of this paper.

E. Markov Decision Processes

A finite MDP is a five-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{G}, \mathcal{R}, \gamma)$, where $\mathcal{S}$ is a finite set of states, $\mathcal{A}$ is a finite set of actions available, $\mathcal{G}$ is the set that contains the MDP transition probabilities given the current state and current action, $\mathcal{R}$ is the set of rewards, and γ is a discount factor.

F. Performance Metric

For any policy π and initial distribution $\mathbf{p}_0$, the finite-horizon reward for a horizon N is the expected discounted total reward [11] defined as follows:

$$v_N^\pi = \mathbb{E}_{\mathbf{p}_0}^\pi \left[\sum_{t=0}^{N-1} \gamma^t R_t(X_t, A_t) + \gamma^N \bar{r}_N(X_N) \right] \tag{4}$$

where $\gamma \in (0,1]$ is a fixed discount factor, which represents the importance of a current reward in comparison to future possible rewards. For a given policy $\pi = \{P_0, \ldots, P_{N-1}\}$, we define the following value function $U_N := \bar{\mathbf{r}}_N$ and

$$U_t := \bar{\mathbf{r}}_{P_t} + \gamma M_{P_t} U_{t+1} \text{ for } t = N-1, \ldots, 0 \tag{5}$$

where

$$\bar{\mathbf{r}}_{P_t} = (R_t \odot P_t) \mathbf{1}. \tag{6}$$

The following lemma establishes the relation among the performance metric, decision policy, value function, and the MC state *pd*.

Lemma 2: The performance metric can be written as follows:

$$v_N^\pi = \sum_{t=0}^{N} \gamma^t \mathbf{p}_t^T \bar{\mathbf{r}}_t \tag{7}$$

$$= \mathbf{p}_0^T U_0 \tag{8}$$

where, for $t = 0 \ldots N-1$, $\bar{\mathbf{r}}_t = \bar{\mathbf{r}}_{P_t}$ is a linear function of the decision matrix P_t given in (6).

Proof: To obtain (7), we apply the linearity property of the expectation to (4) and use the fact that the *pd* of the random variable X_t is the vector $\mathbf{p}_t$ and that the ith component of $\bar{\mathbf{r}}_t$ is $\bar{r}_t(i) = \mathbb{E}[R_t(X_t, A_t) | X_t = i] = \sum_{a \in \mathcal{A}} P_t(i,a) R_t(i,a)$. Equation (8) follows directly from Lemma 1 and (5). ■

Remark 2: Lemmas 1 and 2 show that M_{P_t} and $\bar{\mathbf{r}}_{P_t}$ are linear in the *decision* matrix P_t. This linearity property will be used for *convex synthesis* of decision policies later in this paper.

We assume that the controller knows the transition probabilities $\mathcal{G}$ and the reward function $\mathcal{R}$. The controller can also observe the state X_t, while its *pd* vector $\mathbf{p}_t$ is not available to the controller at any time because $\mathbf{p}_0$ is assumed to be unknown. After observing the state X_t, the system uses a predefined decision matrix P_t as a lookup table to determine the policy action at time t (which is independent of $\mathbf{p}_t$). We also provide algorithms when the controller has access to $\mathbf{p}_t$ and provide policies that are a function of the additional information to improve performance.

IV. Safety Constrained MDP

The optimal policy π^* is defined as the policy that maximizes the performance measure $\pi^* = \text{argmax}_\pi v_N^\pi$, and let v_N^* be the optimal value, i.e., $v_N^* = \max_\pi v_N^\pi$. Note that this maximization is unconstrained and the optimization variables are $P_t(s,a)$ for any $s \in \mathcal{S}$ and $a \in \mathcal{A}$.[2] The *backward induction* (BI) algorithm [11, p. 92], also known as value iteration, is given in Appendix A, and it gives the optimal policy as well as the optimal value. We introduce in this paper the following *safety* constraints into the decision-making policy synthesis:

$$L\mathbf{p}_t \leq \mathbf{d}, \text{ for } t = 1, \ldots, N \tag{9}$$

where $\leq$ denotes the elementwise inequalities, $L \in \mathbb{R}^{q,n}$ is a constant matrix, and $\mathbf{d} \in \mathbb{R}^q$ is a constant vector. The initial state *pd* is assumed to satisfy these constraints, i.e., $L\mathbf{p}_0 \leq \mathbf{d}$. A particular case of these constraints is given by the density upper bound constraints [8], [10] by using $L = I_n$, where I_n is the identity matrix and $\mathbf{d} \in [0,1]^n$. In the multiagent autonomous system example, these constraints can be interpreted as constraints that bound the expected number of agents in a given state at any time epoch.

The standard BI algorithm [11, p. 92] does not take constraints into consideration and cannot be applied to find an optimal decision-making policy when the safety constraints are used. Even finding a feasible policy can be very challenging. We refer to this MDP policy synthesis problem as SC-MDP. The SC-MDP problem with constraints on $\mathbf{p}_t$ can then be written as

$$\begin{aligned} \underset{\pi \in \Pi}{\text{maximize}} \quad & v_N^\pi \\ \text{s.t.} \quad & L\mathbf{p}_t \leq \mathbf{d}, \text{ for } t = 1, \ldots, N \end{aligned} \tag{10}$$

where $\pi = (P_0, \ldots, P_{N-1})$ and $P_t \in \mathbb{R}^{n,p}$ is the matrix of decision variables $P_t(s,a)$. The optimization is over the set of all feasible policies where the decision matrices P_t define *pd*s. Without the safety constraints, the rows of P_t are decoupled optimization variables that are separable in terms of actual constraints: $P_t\mathbf{1} = \mathbf{1}$ and $P_t \geq 0$. With the added set of constraints, which are nonconvex because $\mathbf{p}_t = M_{t-1}^T \ldots M_1^T M_0^T \mathbf{p}_0$, where each of the Markov matrices M_i is a linear function of the variables P_i, the variables in the rows of these decision matrices P_i share common constraints. Hence, the standard BI that leverages on the decoupling between variables at different rows of P_i cannot be applied as is.

Remark 3: Note that the safety constraints given by (9) can be viewed as chance constraints. However, the solution approach in this paper to the safety constraints benefits from the structure of the MDP problem and is fundamentally different from the methods used in standard chance-constraint problems [28]. Standard solution methodology in chance-constrained problems is based on Lagrangian relaxations, which is a general method that works with any form of chance constraints but does not scale well in general with the number of constraints. On the contrary, this paper focuses on the safety constraints on the state *pd* vector that is propagated by the underlying MC $x(k+1) = Mx(k)$, where M is a linear function of the control. The proposed methodology is based on invariance theory for this dynamical system and exploits the problem structure to formulate a convex optimization-based solution approach, which scales favorably with the number of constraints.

Remark 4: The optimal policy π^* of the SC-MDP problem (10) is, in general, a function of the initial state *pd* $\mathbf{p}_0$, i.e., $\pi^* = \pi^*(\mathbf{p}_0)$. However, $\mathbf{p}_0$ may not be known. We show first that without the safety constraints, π^* turns out to be independent of $\mathbf{p}_0$, and this intuitive but not straight-forward result explains why standard unconstrained MDPs have not focused on the assumption of whether $\mathbf{p}_0$ is known or not. However, the synthesis of policies with safety constraints depends on the initial state *pd*.[3] Therefore, the purpose of this paper is to provide SC-MDP feasible policies differentiating the following cases.

1) *Worst-case policy synthesis:* $\mathbf{p}_0$ is not known, and we provide a safe convex set of suboptimal policies characterized by linear inequalities using a worst-case initial distribution analysis.
2) *Robust policy synthesis:* $\mathbf{p}_0$ is known, and the policy is a particular element (this element is a function of $\mathbf{p}_0$) of the suboptimal safe convex set synthesized using the worst-case method.
3) *Forward projection policy synthesis:* $\mathbf{p}_0$ is known, and the policy is a function of $\mathbf{p}_0$, which gives a *safe* policy that is closest (according to a metric defined formally later) to the unconstrained MDP optimal policy. Its safety properties are not robust to perturbations on $\mathbf{p}_0$.

These cases would be elaborated further when the corresponding algorithms are presented.

V. Classical Approach for Constrained MDP

This section introduces the classical approach to constrained MDPs, for which we show that the type of constraints considered does not handle the hard constraints on the transient state *pd* and can only ensure the safety upper bound constraints in the steady state, unlike the methods presented in this paper, which ensure safety for all time epochs. Hence, generalizing the safety

[2] Since v_N^π is continuous in the decision variables that belong to a closed and bounded set, the max is always attained and argmax is well defined.

[3] Note that we only focus on the knowledge of the initial distribution $\mathbf{p}_0$ because $\mathbf{p}_{t+1}$ can be determined from $\mathbf{p}_t$ and P_t using Lemma 1; therefore, $\mathbf{p}_0$ and π deterministically give $\mathbf{p}_1, \ldots, \mathbf{p}_N$.

constraints further to (9) and ensuring their satisfaction during transients is a contribution of this paper.

The classical approach is based on state-action frequencies to address infinite-horizon MDP problems.[4] It formulates a linear program for an MDP [11, p. 224]

$$\underset{y(s,a),s\in\mathcal{S},a\in\mathcal{A}}{\text{maximize}} \quad \sum_s \sum_a R(s,a)y(s,a) \tag{11}$$

$$\text{subject to:} \quad \text{for } i = 1,\dots,n$$

$$\sum_{a\in\mathcal{A}} y(i,a) - \gamma \sum_{s\in\mathcal{S}} \sum_{a\in\mathcal{A}} G_a(s,i)y(s,a) = \mathbf{p}_0(i) \tag{12}$$

$$\sum_{a\in\mathcal{A}} y(i,a) \le d(i) \sum_{s\in\mathcal{S}} \sum_{a\in\mathcal{A}} y(s,a) \tag{13}$$

$$y(i,a) \ge 0, \forall a \in \mathcal{A} \tag{14}$$

where γ is the discount factor[5] for the infinite horizon. Equation (12) is a standard constraint resulting from duality transformation of the optimality equations in the infinite-horizon LP formulation of MDPs with a total discounted reward [11, p. 224]. Equation (13) is the constraint that the classical LP approach supports.

Consider a stationary policy $\pi = (P, P, \dots)$ where the decision matrix P is constructed from a feasible solution of the LP as follows:

$$P(i,k) = \frac{y(i,k)}{\sum_a y(i,a)} \tag{15}$$

then the variable $y(i,a)$ is interpreted as the frequency of performing action a in state s, i.e.,

$$y(i,a) = \sum_{t=0}^{\infty} \gamma^t \text{Prob}[X_t = i, A_t = a|\pi]. \tag{16}$$

The state *pd* $\mathbf{p}_t$ is indirectly related to the variables $y(i,a)$. In fact, $y(i,a)$ can be viewed as a "cost" that is a function of $\mathbf{p}_t, t \ge 0$. To write this function explicitly, we define $\tilde{y}(i)$ to be

$$\tilde{y}(i) = \frac{\sum_a y(i,a)}{\sum_s \sum_a y(s,a)}. \tag{17}$$

Thus, by adding constraints on $\tilde{y}(i)$ in the LP, through (13), and using (16) and (17), we find that $\tilde{y}(i)$ must satisfy

$$\tilde{y}(i) = \frac{\sum_{t=0}^{\infty} \gamma^t \text{Prob}[X_t = i]}{\sum_{t=1}^{\infty} \gamma^t} \tag{18}$$

$$= (1-\gamma) \sum_{t=0}^{\infty} \gamma^t \mathbf{p}_t(i) \tag{19}$$

$$\le d(i). \tag{20}$$

Equation (19) provides important insights relating $\tilde{y}(i)$ to the transient probability of the state *pd* $\mathbf{p}_t(i)$. Note that for a given policy where (9) is satisfied with $L = I$, the second set of constraints in the LP (13) is also satisfied. However, the reverse argument is not always true, i.e., satisfying $\tilde{y}(i) \le d(i)$ does not guarantee that $\mathbf{p}_t(i) \le d(i)$ for all t. In fact, since $\lim_{\gamma\to 1} \tilde{y}(i) = \mathbf{p}_\infty(i)$, where $\mathbf{p}_\infty(i)$ is the stationary distribution of the MC, as $N \to \infty$ and $\gamma \to 1$ in the performance metric (4), the hard constraints are satisfied asymptotically. Even in this case, the classical approach policy gives no guarantees on the transient behavior. Thus, the classical approach cannot be used for the constraints given by (9). As a numerical example, Fig. 1 presents a case where the optimal stationary policy of the classical approach is infeasible for the hard safety constraints, i.e., the state *pd* violates the safety upper bound constraints. Finding optimal policies with hard constraints turns out to be a nonconvex problem, and cannot be solved by the classical LP formulation.

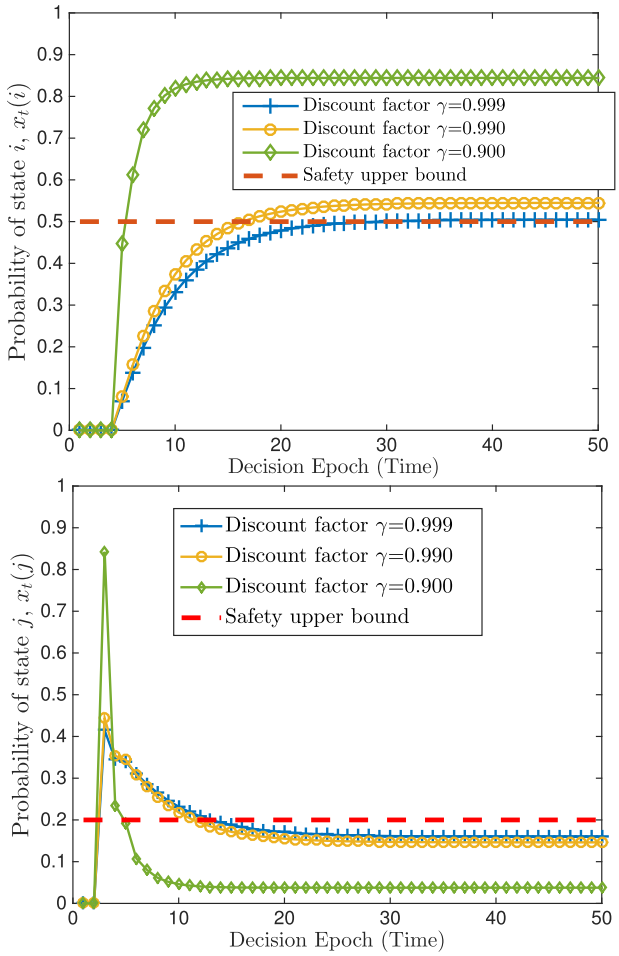


Fig. 1. Dynamics of the probability of a state i (upper plot) and state j (lower plot) in a dynamical system for different values of the discount factor γ. The detailed description of the dynamical system is given in Section VII. The safety upper bound can always be violated independent of the value of γ. As γ approaches 1, the stationary distribution satisfies the safety constraints asymptotically, but at least one state would still violate the constraints during the transient regime.

VI. DP Approach to Markovian Policy Synthesis

In this section, we first provide a new DP formulation of the MDP problem and show that the unconstrained MDP

[4]To account for the finite horizon, the linear program can be tuned by including time as an additional state variable.

[5]For convergence of the LP, γ should be less than 1 (i.e., $\gamma < 1$).

optimal policy π^* is independent of the initial distribution $\mathbf{p}_0$.[6] The optimal policies of the SC-MDP problem, on the other hand, are indeed a function of the initial distribution, and we use the DP formulation of this section to provide *safe* policies that are suboptimal. We further provide policies that enhance the performance when the initial distribution is known to the decision-maker.

The DP formulation works directly with $\mathbf{p}_t$ to be the n-dimensional "state" of the system rather than the actual discrete 1-D state X_t. Thus, the DP n-dimensional state is the MDP state *pd* and should belong to a compact set $\mathcal{P}(\mathcal{S})$, where $\mathcal{P}(\mathcal{S})$ denotes the probability space over the 1-D finite state space $\mathcal{S}$. The dynamics of the DP state $\mathbf{p}_t$ is given by Lemma 1, i.e., $\mathbf{p}_{t+1} = M_{P_t}^T \mathbf{p}_t$. With these dynamics, the state *pd* of the MDP (the DP n-dimensional state) evolves deterministically. A DP policy $\bar{\pi} = (P_0, \ldots, P_{N-1})$ consists of a sequence of functions that map DP states $\mathbf{p}_t$ into controls $P_t = P_t(\mathbf{p}_t)$ such that $P_t(\mathbf{p}_t) \in \mathcal{C}(\mathbf{p}_t)$, where $\mathcal{C}(\mathbf{p}_t)$ is the set of feasible decisions. Since the performance metric can be written using Lemma 2 as $v_N^{\pi} = \sum_{t=0}^{N} \gamma^t \mathbf{p}_t^T \bar{\mathbf{r}}_t$, the *additive reward for stage* t for the DP can be defined as $\mathbf{p}_t^T \bar{\mathbf{r}}_t$. The DP algorithm [31, Proposition 1.3.1, p. 23] given by Algorithm 1 calculates the optimal value v_N^* (and the corresponding optimal policy $\bar{\pi}^*$).

Remark 5: Note that P_t is an optimization variable in $J_{t+1}(M_{P_t}^T \mathbf{p})$ in the algorithm. For a given P_t and $\mathbf{p}$, numerical methods can be used to compute the value of J_{t+1}. But, since P_t itself is an optimization variable, the solution of the optimization problem in line 2 is typically a hard problem. In some special cases, for example, when $J_t(\mathbf{p})$ can be expressed analytically in a closed form, the solution complexity can be reduced significantly, as in unconstrained MDPs.

A. Optimal Policies for Unconstrained MDPs by DP

We use the DP formulation to show that optimal unconstrained MDP policies are independent of the initial distribution $\mathbf{p}_0$. The purpose of this section is to apply the DP algorithm for unconstrained MDPs to extend well-known results of infinite-horizon MDPs to the finite horizon, and set up its use for more complicated SC-MDP problems of this paper.

Without the safety constraints, the set of admissible actions at time t is denoted by $\mathcal{C}(\mathbf{p}_t) = \overline{\mathcal{C}}$, where

$$\overline{\mathcal{C}} = \{P \in \mathbb{R}^{n,p} : P\mathbf{1} = \mathbf{1}, P \geq 0\}. \tag{21}$$

Let $c_i(.)$ be the ith column of a matrix (then $c_i(P^T) = P^T \mathbf{e}_i$ is the vector corresponding to the ith row of the matrix P) and $\mathcal{P}^p$ be a set of probability vectors having dimension $p = |\mathcal{A}|$. Each row of $P \in \overline{\mathcal{C}}$ represents the action choice probabilities for a given state, i.e., $c_i(P^T) \in \mathcal{P}^p$ for $i = 1, \ldots, n$.

Proposition 1: Let $\mathcal{C}(\mathbf{p}_t) = \overline{\mathcal{C}}$, where $\overline{\mathcal{C}}$ is given by (21). Then, the optimal policy $\bar{\pi}^*(\mathbf{p}_0) = \{P_0^*(\mathbf{p}_0), \ldots, P_{N-1}^*(\mathbf{p}_{N-1})\}$ resulting from Algorithm 1 is independent from $\mathbf{p}_0$.

[6]While this result is known for infinite-horizon problems, see, for example, [11, Proposition 7.2.19(b), p. 301] for a proof based on the LP formulation of optimal MDP policies, we provide here a novel simpler proof for *finite-horizon* problems using the novel DP formulation.

Algorithm 1: Dynamic Programming.

1: Start with $J_N(\mathbf{p}) = \mathbf{p}^T \bar{\mathbf{r}}_N$
2: For $t = N-1, \ldots, 0$

$$J_t(\mathbf{p}) = \max_{P_t \in \mathcal{C}(\mathbf{p})} \left\{ \mathbf{p}^T \bar{\mathbf{r}}_{P_t} + \gamma J_{t+1}(M_{P_t}^T \mathbf{p}) \right\}.$$

3: **Result:** $J_0(\mathbf{p}) = v_N^*$.

Proof: We first mention that even if the set of feasible policies $\mathcal{C}(\mathbf{p}_t)$ is independent of $\mathbf{p}_t$; in general, this does not guarantee that the optimal decision matrix $P_t^*(\mathbf{p}_t)$ is independent of $\mathbf{p}_t$, so $\bar{\pi}^*$ might be a function of $\mathbf{p}_0$. However, we will show by this proposition that $\bar{\pi}^*$ is indeed independent of $\mathbf{p}_0$.

The proof proceeds by induction to show that, in the DP Algorithm 1, $J_t(\mathbf{p})$ has the following closed-form expression: $J_t(\mathbf{p}) = \mathbf{p}^T V_t^*$, where V_t^* is the vector of optimal value function at every state computed by Algorithm 6 in the Appendix. The base case for $t = N$ holds true by construction, i.e., $J_N(\mathbf{p}) = \mathbf{p}^T \bar{\mathbf{r}}_N = \mathbf{p}^T V_N^*$. Iterating backward from $N-1$ to 0, suppose now the hypothesis holds true for $t+1$ (i.e., $J_{t+1}(\mathbf{p}) = \mathbf{p}^T V_{t+1}^*$), we show it is true for t

$$J_t(\mathbf{p}) = \max_{P_t \in \overline{\mathcal{C}}} \{\mathbf{p}^T \bar{\mathbf{r}}_t + \gamma J_{t+1}(M_t^T \mathbf{p})\} \tag{22}$$

$$= \max_{c_i(P^T) \in \mathcal{P}^p, i \in \mathcal{S}} \{\mathbf{p}^T \bar{\mathbf{r}}_t + \gamma \mathbf{p}^T M_t V_{t+1}^*\} \tag{23}$$

$$= \max_{c_i(P^T) \in \mathcal{P}^p, i \in \mathcal{S}} \left\{ \sum_i \mathbf{p}(i) (\bar{r}_t(i) + \gamma \mathbf{e}_i^T M_t V_{t+1}^*) \right\} \tag{24}$$

$$= \sum_i \mathbf{p}(i) \left(\max_{c_i(P^T) \in \mathcal{P}^p} \{\bar{r}_t(i) + \gamma \mathbf{e}_i^T M_t V_{t+1}^*\} \right) \tag{25}$$

where (23) uses the induction hypothesis and that each row of P_t is a probability vector. Equation (25) is due to the fact that $\mathbf{p}(i) \geq 0$ for all i and the value function is separable in terms of the optimization variables because $\bar{r}_t(i)$ and $\mathbf{e}_i^T M_t$ are both a function of variables in the ith row of P_t as shown in the proof to Lemma 2 for $\bar{r}_t(i)$ and in (1) for $\mathbf{e}_i^T M_t$. Since we have a finite number of actions, the maximization inside the parenthesis in the last equation can be written as follows:

$$\max_{c_i(P^T) \in \mathcal{P}^m} \{\bar{r}_t(i) + \gamma \mathbf{e}_i^T M_t V_{t+1}^*\}$$

$$= \max_k \left\{ R_t(i,k) + \gamma \sum_{j \in \mathcal{S}} G_{t,k}(i,j) V_{t+1}^*(j) \right\}$$

$$= V_t^*(i)$$

and hence $J_t(\mathbf{p}) = \sum_i p(i) V_t^*(i) = \mathbf{p}^T V_t^*$, which has a closed-form solution as function of $\mathbf{p}$. This justifies that the calculation of V_t^* for $t = N, \ldots, 0$ in Algorithm 6 (in the Appendix) is sufficient for finding the optimal value of the MDP given by $v_N^* = J_0(\mathbf{p}_0) = \mathbf{p}_0^T V_0^*$, and since V_t^* is independent of $\mathbf{p}_t$, so is P_t^*; thus, the optimal policy $\bar{\pi}^*$ is independent of $\mathbf{p}_0$ and this completes the proof. ■

Algorithm 2: SC-MDP Worst Case.

1: **Definitions:** P_t for $t = 0, ..., N-1$ are the optimization variables, describing the decision policy. Let $\mathcal{X}$ be the set of safe *pd*s and let $\hat{\mathcal{C}}$ be the set of safe policies.
2: Set $\hat{U}_N = \bar{\mathbf{r}}_N$.
3: For $t = N-1, \ldots, 0$, given $\hat{U}_{t+1}$, compute a policy $\hat{P}_t \in \hat{\mathcal{C}}_t$, where $\hat{\mathcal{C}}_t$ is a convex subset of $\hat{\mathcal{C}}$ having *worst-case* suboptimal policies defined as follows:

$$\hat{\mathcal{C}}_t := \underset{P_t \in \hat{\mathcal{C}}}{\operatorname{argmax}} \min_{\mathbf{p} \in \mathcal{X}} \mathbf{p}^T U_{P_t} \tag{26}$$

where the value function vector U_{P_t} is

$$U_{P_t} = \bar{\mathbf{r}}_{P_t} + \gamma M_{P_t} \hat{U}_{t+1} \tag{27}$$

and $\hat{U}_t = U_{\hat{P}_t}$ for a computed policy $\hat{P}_t \in \hat{\mathcal{C}}_t$.
4: **Result:** $v_N^* \geq \mathbf{p}_0^T \hat{U}_0$

Remark 6: V_1^* obtained via Algorithm 6 leads to a *deterministic Markovian* policy, which defines an optimal policy for the unconstrained MDP, i.e., the policy that maximizes the total expected reward. It must be emphasized that if state constraints were present, this policy can be infeasible. The optimal SC-MDP policy would then be a function of the initial distribution $\mathbf{p}_0$.

B. SC-MDP Worst-Case Policy Synthesis by DP

In this section, $\mathbf{p}_0$ is assumed to be unknown to the decision-maker. When the state constraints are present, $J_t(\mathbf{p})$ does not have a closed-form solution, and hence finding an optimal (even a feasible) solution is challenging. This section presents a new algorithm, Algorithm 2, to compute a feasible solution of the state constrained MDPs with lower bound guarantees on the expected reward. Then, we provide a linear program to implement this algorithm efficiently. We first define two terms related to the safety constraints that are used in the following subsections. Let $\mathcal{X}$ be the set of safe *pd*'s defined as follows:

$$\mathcal{X} = \{\mathbf{p} \in \mathbb{R}^n : 0 \leq L\mathbf{p} \leq \mathbf{d},\ \mathbf{1}^T\mathbf{p} = 1, \mathbf{p} \geq 0\}$$

and let $\hat{\mathcal{C}}$ be the set of safe policies defined as follows:

$$\hat{\mathcal{C}} = \bigcap_{\mathbf{p} \in \mathcal{X}} \mathcal{C}(\mathbf{p})$$

with $\mathcal{C}(\mathbf{p}) = \{P \in \mathbb{R}^{n,p} : P\mathbf{1} = \mathbf{1},\ P \geq 0, LM_P^T\mathbf{p} \leq \mathbf{d}\}$, where $L \in \mathbb{R}^{q,n}$ and $\mathbf{d} \in \mathbb{R}^q$. M_P is given by (3). We assume that $\hat{\mathcal{C}}$ is nonempty, i.e., there exists at least one safe policy $P \in \hat{\mathcal{C}}$.

1) Feasible Suboptimal Policies: Algorithm 2 establishes an SC-MDP feasible policy that is independent of initial distribution by considering a worst-case study.

Remark 7: The optimal solution of (26), in general, is not unique; hence, "argmax" is set-valued. Among the possible solution variables $\hat{P}_t \in \hat{\mathcal{C}}_t$, we may be interested in decision policies that are as close as possible to P_t^* (the unconstrained optimal policy computed by Algorithm 6 at iteration t) because the latter gives the maximum total expected reward for an unconstrained MDP problem but might not be feasible due to the safety state constraints. Therefore, from a practical standpoint, we can target the projection of P_t^* on $\hat{\mathcal{C}}_t$. We can choose $\hat{P}_t$ to be the solution of the following optimization:

$$\hat{P}_t = \underset{P \in \hat{\mathcal{C}}_t}{\operatorname{argmin}} \quad ||P - P_t^*||$$

where $||.||$ can be any matrix norm.[7] Note that if $P_t^* \in \hat{\mathcal{C}}$, then $P_t^* \in \hat{\mathcal{C}}_t$ and this optimization will give the optimal policy. Therefore, this extra optimization retrieves back the solution of the unconstrained MDP if the state constraints were relaxed.

Theorem 1: Algorithm 2 provides a feasible policy $\hat{\pi} = \{\hat{P}_0, \ldots, \hat{P}_{N-1}\}$ for the SC-MDP problem (10), which guarantees the total expected reward to be greater than $R^{\#} = \min_{\mathbf{p} \in \mathcal{X}} \mathbf{p}^T \hat{U}_0$, i.e., $v_N^* \geq R^{\#}$, where $\hat{U}_0 = U_{\hat{P}_0}$ is given by the recursion (27).

Proof: The proof is based on applying constraints to the DP Algorithm 1.[8] Letting $J_N(\mathbf{p}) = \mathbf{p}^T \bar{\mathbf{r}}_N$, it will be shown by induction that $J_t(\mathbf{p}) \geq \mathbf{p}^T \hat{U}_t$. It is true for $t = N$. Now supposing that it is true for $t+1$, let us prove it true for t. From Algorithm 1, we have

$$J_t(\mathbf{p}) = \max_{P_t \in \mathcal{C}(\mathbf{p})} \left\{\mathbf{p}^T \bar{\mathbf{r}}_{P_t} + \gamma J_{t+1}(M_{P_t}^T \mathbf{p})\right\} \tag{28}$$

$$\geq \max_{P_t \in \mathcal{C}(\mathbf{p})} \left\{\mathbf{p}^T \bar{\mathbf{r}}_{P_t} + \gamma \mathbf{p}^T M_{P_t} \hat{U}_{t+1}\right\} \tag{29}$$

$$\geq \max_{P_t \in \hat{\mathcal{C}}} \left\{\mathbf{p}^T (\bar{\mathbf{r}}_{P_t} + \gamma M_{P_t} \hat{U}_{t+1})\right\} \tag{30}$$

$$\geq \mathbf{p}^T \hat{U}_t \tag{31}$$

where in the second line we applied the induction hypothesis and the last line follows from the definition of $\hat{U}_t$ in (27). Therefore

$$\begin{aligned} v_N^* &= J_0(\mathbf{p}_0) \\ &\geq \mathbf{p}_0^T \hat{U}_0 \\ &\geq \min_{\mathbf{p} \in \mathcal{X}} \mathbf{p}^T \hat{U}_0 = R^{\#} \end{aligned}$$

where the first equality comes from line 3 in Algorithm 1, and this completes the proof. ■

2) LP Formulation of $\hat{P}_t$: The calculation of $\hat{\mathcal{C}}_t$ for $t = N-1, \ldots, 0$ in (26) is the main challenge in the implementation of Algorithm 2. This section describes an LP-approach to the computation of a solution $\hat{P}_t \in \hat{\mathcal{C}}_t$ in every iteration loop of Step 3 in Algorithm 2. From the algorithm, $\hat{\mathcal{C}}_t$ is given as follows:

$$\hat{\mathcal{C}}_t := \underset{P \in \hat{\mathcal{C}}}{\operatorname{argmax}} \min_{\mathbf{p} \in \mathcal{X}} \mathbf{p}^T U_P$$

$$\text{where } U_P = \bar{\mathbf{r}}_P + \gamma M_P \hat{U}_{t+1}$$

$$\text{with } \hat{U}_N = \bar{\mathbf{r}}_N \text{ and } \hat{U}_t = U_{\hat{P}_t},\ t = N-1, \ldots, 0. \tag{32}$$

[7] We considered the Frobenius norm in the simulations.

[8] Note that the proof techniques of including constraints within the DP iteration used in this paper are new and can be of independent interest for the reader.

As given earlier by Lemmas 1 and 2, we have

$$M_P = \sum_k G_{t,k} \odot \left((P\mathbf{e}_k)\mathbf{1}^T\right) \text{ and } \bar{\mathbf{r}}_P = (R_t \odot P)\mathbf{1}. \quad (33)$$

Note that $\hat{\mathcal{C}}_t$ is convex because it is the "argmax" of maximizers of a concave function $f(P) = \min_{\mathbf{p}\in\mathcal{X}} \mathbf{p}^T U_P$ on a convex domain $\hat{\mathcal{C}}$. The following theorem provides a convex characterization of $\hat{\mathcal{C}}_t$ and an LP formulation for the computation of $\hat{P}_t$.

Theorem 2: The max–min problem given by (32) with (33) can be solved by the following equivalent LP problem (given t, L, $\mathbf{d}$, $G_{t,k}$ for $k = 1 \dots p$, R_t, and $\hat{U}_{t+1}$):

$$\begin{aligned} \underset{P,\mathbf{y},z,\mathbf{r},M,S,\mathbf{s},K}{\text{maximize}} \quad & -\mathbf{d}^T\mathbf{y} + z \\ \text{subject to} \quad & M = \sum_{k=1}^{p} G_{t,k} \odot \left((P\mathbf{e}_k)\mathbf{1}^T\right) \\ & \mathbf{r} = (R_t \odot P)\mathbf{1} \\ & -L^T\mathbf{y} + z\mathbf{1} \le \mathbf{r} + \gamma M\hat{U}_{t+1} \\ & KL = LM^T + S + \mathbf{s}\mathbf{1}^T \\ & \mathbf{s} + \mathbf{d} \ge K\mathbf{d} \\ & P\mathbf{1} = \mathbf{1}, P \ge 0, \mathbf{y} \ge 0, S \ge 0, K \ge 0. \end{aligned} \quad (34)$$

Proof: The proof will use the duality theory of LP, which uses the fact that the following primal and dual problems produce the same cost:

PRIMAL	DUAL
minimize $\mathbf{b}^T\mathbf{p}$	maximize $\mathbf{c}^T\mathbf{y}$
s.t. $A^T\mathbf{p} \ge \mathbf{c}, \mathbf{p} \ge 0$	s.t. $A\mathbf{y} \le \mathbf{b}, \mathbf{y} \ge 0$.

Since the set $\mathcal{X}$ is defined as $0 \le L\mathbf{p} \le \mathbf{d}$ and $\mathbf{p}^T\mathbf{1} = 1$, the min in (32) can be obtained by a minimization problem with the following primal problem parameters:

$$\mathbf{b} = U_P, A = [-L^T \ \mathbf{1} \ -\mathbf{1}], \text{ and } \mathbf{c}^T = [-\mathbf{d}^T \ 1 \ -1].$$

The dual of this program is

$$\begin{aligned} \underset{\mathbf{y},z}{\text{maximize}} \quad & -\mathbf{d}^T\mathbf{y} + z \\ \text{subject to} \quad & -L^T\mathbf{y} + z\mathbf{1} \le U_P \\ & \mathbf{y} \ge 0, z \text{ unconstrained} \end{aligned}$$

where $U_P = \mathbf{r}_P + \gamma M_P\hat{U}_{t+1}$. Notice that the above formulation provides an important part of the structure of the LP in the theorem. Next, by considering the argmax in (32), it remains to show that the set $\hat{\mathcal{C}}$ can be represented by linear inequalities to write (32) as a maximization LP problem. Let $\mathcal{M} = \cap_{\mathbf{p}\in\mathcal{X}} \mathcal{M}(\mathbf{p})$, where $\mathcal{M}(\mathbf{p}) = \{M \in \mathbb{R}^{n,n}, M\mathbf{1} = \mathbf{1}, M \ge 0, LM^T\mathbf{p} \le \mathbf{d}\}$, then

$$\begin{aligned} M \in \mathcal{M} \quad \Leftrightarrow \quad & \exists S \ge 0, K \ge 0, \mathbf{s} \text{ such that} \\ & KL = LM^T + S + \mathbf{s}\mathbf{1}^T \\ & \mathbf{s} + \mathbf{d} \ge K\mathbf{d} \\ & M\mathbf{1} = \mathbf{1}, M \ge 0. \end{aligned} \quad (35)$$

See [10, Th. 1] for a proof of the above equivalence with a small modification to account for the matrix L because $L = I$ is an identity matrix in [10, Th. 1]. Using Lemma 1, M is a linear function of the decision variable P, and the set $\hat{\mathcal{C}}$ is equivalently described by $\hat{\mathcal{C}} = \{P \in \mathbb{R}^{n,p}, P\mathbf{1} = \mathbf{1}, P \ge 0, M_P \in \mathcal{M}\}$, which implies that $\hat{\mathcal{C}}$ can be described by linear inequalities

$$KL = LM^T + S + \mathbf{s}\mathbf{1}^T \quad (36)$$

$$\mathbf{s} + \mathbf{d} \ge K\mathbf{d} \quad (37)$$

$$S \ge 0, K \ge 0 \quad (38)$$

$$M = \sum_{k=1}^{p} G_{t,k} \odot \left((P\mathbf{e}_k)\mathbf{1}^T\right) \quad (39)$$

$$P\mathbf{1} = \mathbf{1}, P \ge 0 \quad (40)$$

where (36)–(38) come from (35), (39) is due to Lemma 1, and (40) is because P is a decision matrix that defines probability mass functions over actions. Now combining this result with the dual program, we can conclude that $\hat{P}_t$ can be obtained via the linear program given in the theorem, which concludes the proof. ■

Remark 8: Note that due to the assumption that $\hat{\mathcal{C}}$ is nonempty, the LP in Theorem 2 always has a feasible solution. For the general case, if the LP is infeasible, the safety constraints characterized by the parameters L and $\mathbf{d}$ can be relaxed (e.g., increasing the values in $\mathbf{d}$) to ensure feasibility.

The worst-case policies are independent of initial values. Hence, since optimal SC-MDP policies are a function of $\mathbf{p}_0$, these policies are suboptimal.

C. SC-MDP Robust Policy Synthesis

In this section, $\mathbf{p}_0$ is assumed to be known to the decision-maker. In Algorithm 2, Step 3, the expression of $\hat{P}_t$

$$\hat{P}_t \in \underset{P_t\in\hat{\mathcal{C}}}{\text{argmax}} \ \mathbf{p}_{\min}(P_t)^T \left(\bar{\mathbf{r}}_{P_t} + \gamma M_{P_t}\hat{U}_{t+1}\right) \quad (41)$$

seeks a solution for P_t based on the worst-case density distribution, i.e.,

$$\mathbf{p}_{\min}(P_t) \in \underset{\mathbf{p}\in\mathcal{X}}{\text{argmin}} \ \mathbf{p}^T \left(\bar{\mathbf{r}}_{P_t} + \gamma M_{P_t}\hat{U}_{t+1}\right).$$

Clearly, this is a conservative approach because we can do the same optimization procedure with any density $\mathbf{p}$ replacing $\mathbf{p}_{\min}$ in (43). In fact, the closer the actual $\mathbf{p}_t$ to the one used in the optimization equation, the better the policy synthesized approximates the optimal policy. Unfortunately, even with the knowledge of $\mathbf{p}_0$, having a good approximation of $\mathbf{p}_t$ is not easy because it is a function of the "yet to be calculated" decision matrices for the stages $0, \dots, t-1$. Adding these decision matrices

Algorithm 3: SC-MDP Robust: Backward–Forward Induction.

1: Start with an initial policy $\hat{\pi}^0 = \{P_0^0, \ldots, P_{N-1}^0\}$, an initial density distribution $\mathbf{p}_0$, an initial value function $v^0 = R^{\#}$, specify $\epsilon > 0$, and set iteration $k = 0$.

2: Let $\hat{\mathbf{p}}_0 = \mathbf{p}_0$, and $\hat{\mathbf{p}}_{t+1} = M_{P_t^0}^T \hat{\mathbf{p}}_t$, for $t = 0, \ldots, N-1$

3: **Backward Induction (BI):**
Let $\hat{U}_N = \bar{\mathbf{r}}_N$
For $t = N-1, \ldots, 0$, given $\hat{U}_{t+1}$ and $\hat{\mathbf{p}}_t$, compute the policy

$$\hat{P}_t \in \underset{P_t \in \hat{\mathcal{C}}_t}{\operatorname{argmax}} \ \hat{\mathbf{p}}_t^T \left(\bar{\mathbf{r}}_{P_t} + \gamma M_{P_t} \hat{U}_{t+1} \right)$$

and the vector of expected discounted sum of reward

$$\hat{U}_t = \bar{\mathbf{r}}_{\hat{P}_t} + \gamma M_{\hat{P}_t} \hat{U}_{t+1}$$

4: **Forward Induction (FI):**
Given $\hat{\mathbf{p}}_0$ and $\{\hat{P}_t, t = 0, \ldots, N-1\}$, recompute density distribution according to the updated policy

$$\hat{\mathbf{p}}_{t+1} = M_{\hat{P}_t}^T \hat{\mathbf{p}}_t, \text{ for } t = 0, \ldots, N-1$$

5: **Stopping Condition:**

$$v^{k+1} = \max \left\{ v^k, \sum_{t=0}^{N} \gamma^t \hat{\mathbf{p}}_t^T \bar{\mathbf{r}}_t \right\}$$

$$\hat{\pi}^{k+1} = \underset{\hat{\pi}}{\operatorname{argmax}} \left\{ v^k, \sum_{t=0}^{N} \gamma^t \hat{\mathbf{p}}_t^T \bar{\mathbf{r}}_t \right\}$$

Stop if $|v^{k+1} - v^k| < \epsilon$ and return the policy $\hat{\pi}^{k+1}$. Otherwise increment k by 1 and return to Step 3

in the optimization (43) renders the problem nonconvex due to the multilinearities that arise. To reduce the conservatism while keeping the computational efficiency tractable, we consider a better estimate of density $\mathbf{p}$ instead of $\mathbf{p}_{\min}$ with Algorithm 3.

Algorithm 3 terminates when v^k converges, and the algorithm outputs a policy whose total expected reward does not improve by more than ϵ (a user-defined quantity). The convergence of v^k is ensured by the following lemma.

Lemma 3: The sequence $\{v^k\}$ generated by Algorithm 3 is nondecreasing and bounded from above. Hence, for any $\epsilon > 0$, the algorithm converges after a finite number of iterations.

Proof: v^k is nondecreasing by construction

$$v^{k+1} = \max \left\{ v^k, \sum_{t=0}^{N} \gamma^t \hat{\mathbf{p}}_t^T \bar{\mathbf{r}}_t \right\} \geq v^k.$$

v^k is upper bounded because

$$\sum_{t=0}^{N} \gamma^t \hat{\mathbf{p}}_t^T \bar{\mathbf{r}}_t \leq \sum_{t=0}^{N} \gamma^t \hat{\mathbf{p}}_t^T (r_{\max} \mathbf{1})$$

$$= r_{\max} \sum_{t=0}^{N} \gamma^t = r_{\max} \frac{(1 - \gamma^{N+1})}{(1 - \gamma)}$$

where $r_{\max} = \max_{i,t} \bar{\mathbf{r}}_t(i)$. Consequently, there is a finite limit of the sequence. Hence, given $\epsilon > 0$, there exists an integer $K > 0$ such that $|v_{k+1} - v_k| < \epsilon$ for $k > K$. ■

The simulations show that the algorithm indeed converges after only few backward–forward iterations. Moreover, the policy is synthesized in an iterative manner where starting from a given initial distribution $\mathbf{p}_0$ and a worst-case policy from Algorithm 2 $\{P_0^0, \ldots, P_{N-1}^0\}$, Algorithm 3 is based on two steps of iterations: forward and backward. In the forward induction step, given the initial distribution $\mathbf{p}_0$ and P_t^0, we first calculate an estimate of the state *pd* at time t using Lemma 1: $\{\hat{\mathbf{p}}_t, t = 1, \ldots, N\}$. In the BI step, we calculate $\hat{P}_t$ for $t = 1, \ldots, N-1$ with

$$\hat{P}_t \in \underset{P_t \in \hat{\mathcal{C}}_t}{\operatorname{argmax}} \ \hat{\mathbf{p}}_t^T \left(\bar{\mathbf{r}}_{P_t} + \gamma M_{P_t} \hat{U}_{t+1} \right). \tag{42}$$

Remark 9: The feasible region of policies in the BI equation (44) of Algorithm 3 is $\hat{\mathcal{C}}_t$: the same policy set defined by (26) in Algorithm 2. By limiting the search region of policies to the set $\hat{\mathcal{C}}_t$, Algorithm 3 outputs a policy that is also a feasible policy in Algorithm 2; thus, it also guarantees the lower bound performance given by Theorem 1.

Analogous to Theorem 2, (44) can be formulated as a linear program. Let us define v_{opt} to be the optimal value of problem (42), calculated simply by running the corresponding LP. Then, we have $P \in \hat{\mathcal{C}}_t$ if and only if there exist variables that satisfy the constraints in (42) and the additional constraint $-\mathbf{d}^T y + z \geq v_{\text{opt}}$. Let $\tilde{U}_{t+1}$ be the value function that is defined on the initial policy, i.e., $\tilde{U}_N = \bar{\mathbf{r}}_N$ and

$$\tilde{U}_t = \bar{\mathbf{r}}_{P_t^0} + \gamma M_{P_t^0} \tilde{U}_{t+1} \text{ for } t = N-1, \ldots, 0.$$

Then, the linear program is as follows:

$$\underset{P_t, \mathbf{y}, z, \mathbf{r}, M, S, \mathbf{s}, K}{\text{maximize}} \quad \hat{\mathbf{p}}_t^T \left(\mathbf{r} + \gamma M \hat{U}_{t+1} \right) \tag{43}$$

$$\text{subject to} \quad -\mathbf{d}^T y + z \geq v_{\text{opt}} \tag{44}$$

$$M = \sum_{k=1}^{p} G_{t,k} \odot \left((P_t \cdot \mathbf{e}_k) \mathbf{1}^T \right) \tag{45}$$

$$\mathbf{r} = (R_t \odot P_t) \mathbf{1} \tag{46}$$

$$-L^T \mathbf{y} + z\mathbf{1} \leq \mathbf{r} + \gamma M \tilde{U}_{t+1} \tag{47}$$

$$KL = LM^T + S + \mathbf{s}\mathbf{1}^T \tag{48}$$

$$\mathbf{s} + \mathbf{d} \geq K\mathbf{d} \tag{49}$$

$$P_t \mathbf{1} = \mathbf{1}, P_t \geq 0, \mathbf{y} \geq 0, S \geq 0, K \geq 0. \tag{50}$$

Constraints (47)–(52) are the same constraints used in the LP (42), and adding constraint (46) to those would guarantee that $P_t \in \hat{\mathcal{C}}_t$.

The policy of "SC-MDP Robust" algorithm is selected from the set of worst-case policies. Thus, it gives a suboptimal performance for any $\mathbf{p}_t, t = 0, \ldots, N$. Even after some perturbations on the value of $\hat{\mathbf{p}}_t$ due to some noise in the system, as long as the policy remains in the safe set $\mathcal{X} = \{\mathbf{p} \in \mathbb{R}^n : 0 \leq L\mathbf{p} \leq \mathbf{d}, \mathbf{1}^T \mathbf{p} = 1\}$, the policy would still provide a suboptimal performance, hence the robustness properties. Next, we present

Algorithm 4: SC-MDP Forward Projection.

1: Let $V_N^* = \bar{\mathbf{r}}_N$, for $t = N-1, \ldots, 1 \quad \forall s \in \mathcal{S}$,

$$V_t^*(s) = \max_{a \in \mathcal{A}} \left[R_t(s,a) + \gamma \sum_{j \in \mathcal{S}} G_{t,a}(j,s) V_{t+1}^*(j) \right]$$

Note that $\{V_t^*, t = 1, \ldots, N\}$ are the same value functions as for the optimal unconstrained MDP (values calculated in Algorithm 6 in the Appendix)

2: Given $\mathbf{p}_0$, for $t = 0, 1, \ldots, N-1$, compute policy

$$\hat{P}_t \in \underset{P_t \in \mathcal{C}(\mathbf{p}_t)}{\operatorname{argmax}} \mathbf{p}_t^T (\bar{\mathbf{r}}_{P_t} + \gamma M_{P_t} V_{t+1}^*) \tag{51}$$

and

$$\mathbf{p}_{t+1} = M_{\hat{P}_t}^T \mathbf{p}_t$$

an alternative algorithm—open-loop policy synthesis—that is a projection of the unconstrained optimal MDP policy on a set that satisfies safety constraints.

D. Forward Projection Policy Synthesis

Also in this section, $\mathbf{p}_0$ is assumed to be known to the decision-maker. As noted earlier, the main challenge in the implementation of the DP for the SC-MDP problem is the bilinearity between the value function U_t and the state *pd* vector $\mathbf{p}_t$ at every stage of the problem (29). The less strict the constraints are, the closer the optimal value function U_t would be to the unconstrained optimal value function V_t^*. Thus, we can use V_t^* as an approximation of U_t to eliminate the bilinearity, and thus, we can use a *forward induction* procedure for a given $\mathbf{p}_0$ to synthesize a *safe* policy that is closest to the unconstrained (possibly unsafe) optimal policy. Algorithm 4 uses *forward induction* by estimating U_t with the corresponding value function V_t^* of unconstrained MDP. It is based on the assumption that $\mathbf{p}_0$ is known. The "SC-MDP Forward Projection" policy synthesis algorithm is given by Algorithm 4.

To further study the theoretical properties of the resulting policy, let us define the matrix $B \in \mathbb{R}^{n,np}$ as a matrix that satisfies the following equation:

$$\bar{\mathbf{r}}_P + \gamma M_P V = B \operatorname{vect}(P) \tag{52}$$

where vect(.) stacks the rows of a matrix in a vector and $V \in \mathbb{R}^n$ is a given vector. In particular, by using the expressions of M_P and $\bar{\mathbf{r}}_P$ from Lemmas 1 and 2, we can write the closed-form expressions of the nonzero entries of B as follows: For $i = 1, \ldots, n$ and $k = 1, \ldots, p$, we have

$$B(i, (i-1)p + k) = R_t(i,k) + \gamma \sum_{s \in \mathcal{S}} G_{t,k}(i,s) V(s).$$

The following proposition shows that the policy given by Algorithm 4 is a projection of the unconstrained MDP optimal policy on the set of safe policies.

Proposition 2: Let $\hat{P}_t$ be the decision matrix constructed by Algorithm 4, then

$$\hat{P}_t \in \underset{P \in \mathcal{C}(\mathbf{p}_t)}{\operatorname{argmin}} ||B(\operatorname{vect}(P_t^*) - \operatorname{vect}(P))||_{\mathbf{p}_t, 1} \tag{53}$$

where P_t^* is an optimal unconstrained MDP policy given by Algorithm 6 in the Appendix, B is the matrix given in (54), $||.||_{\mathbf{p}_t,1}$ is the weighted L_1 seminorm, weighted by the non-negative vector $\mathbf{p}_t$, and $\mathcal{C}(\mathbf{p}_t) = \{P \in \mathbb{R}^{n,p} : P\mathbf{1} = \mathbf{1}, P \geq 0, LM_P^T \mathbf{p}_t \leq \mathbf{d}\}$.

Proof: By considering (53) in the algorithm, we have

$$\mathcal{P}_t := \underset{P_t \in \mathcal{C}(\mathbf{p}_t)}{\operatorname{argmax}} \mathbf{p}_t^T (\bar{\mathbf{r}}_{P_t} + \gamma M_{P_t} V_{t+1}^*).$$

The admissible set due to the "argmax" in the above expression would not change if we used "argmin" and multiply the objective function by -1; then, the above expression is

$$\mathcal{P}_t = \underset{P_t \in \mathcal{C}(\mathbf{p}_t)}{\operatorname{argmin}} - \mathbf{p}_t^T (\bar{\mathbf{r}}_{P_t} + \gamma M_{P_t} V_{t+1}^*).$$

We can also add any term independent of P_t to the above expression, e.g., $\mathbf{p}_t^T V_t^*$

$$\mathcal{P}_t = \underset{P_t \in \mathcal{C}(\mathbf{p}_t)}{\operatorname{argmin}} \mathbf{p}_t^T \left(V_t^* - (\bar{\mathbf{r}}_{P_t} + \gamma M_{P_t} V_{t+1}^*) \right).$$

Since $V_t^* = \bar{\mathbf{r}}_{P_t^*} + \gamma M_{P_t^*} V_{t+1}^*$, and using (54), the above expression reduces to

$$\begin{aligned} \mathcal{P}_t &= \underset{P_t \in \mathcal{C}(\mathbf{p}_t)}{\operatorname{argmin}} \mathbf{p}_t^T \left(B(\operatorname{vect}(P_t^*) - \operatorname{vect}(P)) \right) \\ &= \underset{P_t \in \mathcal{C}(\mathbf{p}_t)}{\operatorname{argmin}} ||B(\operatorname{vect}(P_t^*) - \operatorname{vect}(P))||_{\mathbf{p}_t, 1} \end{aligned}$$

where the last equality holds since $V_t^* \geq \bar{\mathbf{r}}_{P_t} + \gamma M_{P_t} V_{t+1}^*$ for all $P_t \in \mathcal{C}(\mathbf{p}_t)$. ■

Proposition 2 shows that if an unconstrained optimal policy P_t^* is feasible (e.g., if $L = I_n$ and $\mathbf{d} = \mathbf{1}$), then $\hat{P}_t = P_t^*$. An advantage of this approach is that it requires less computational complexity than other algorithms given in this section.

E. Extension of Algorithm 2 to Infinite Horizon

While the focus of this paper is on finite horizon, the SC-MDP algorithms can also be used for infinite horizon. We consider in this section standard assumptions for the infinite horizon.

1) Stationary MDP transition probabilities, i.e., $G_{t,k}(i,j) = G_k(i,j)$ for all t.
2) Stationary reward function, i.e., $R_t(s,a) = R(s,a)$ for all t.
3) Stationary Markovian policies, i.e., $P_t = P$ for all t and $\pi = \{P, P, \ldots\}$.

For unconstrained MDPs, it is well known that there exists an optimal policy maximizing the infinite-horizon total discounted reward function ($v^\pi = \mathbb{E}_{\mathbf{p}_0}^\pi [\sum_{t=0}^\infty \gamma^t R(X_t, A_t)]$) that is stationary and Markovian. In the presence of safety constraints (9), optimal policies may not be stationary, as shown in [16]. However, we still restrict our analysis to stationary policies because they can be efficiently computed using an LP.

Algorithm 5: Infinite Horizon: SC-MDP.

1: Select $\hat{V}^0(s) = 0$ for all $s \in \mathcal{S}$, specify $\epsilon > 0$, and set $t = 0$

2: For each $s \in \mathcal{S}$, compute $\hat{V}^{t+1}(s)$ by

$$\hat{V}^{t+1}(s) = \max_{P \in \hat{C}} \left\{ \bar{r}_P(s) + \gamma \sum_{j \in \mathcal{S}} M_P(s,j) \hat{V}^t(j) \right\}$$

3: If $||\hat{V}^{t+1} - \hat{V}^t|| < \epsilon$, go to Step 4. Otherwise increment t by 1 and return to Step 2

4: Choose the stationary policy $\hat{\pi} = \{\hat{P}, \hat{P}, \dots\}$, where

$$\hat{P} \in \operatorname*{argmax}_{P \in \hat{\mathcal{C}}} \min_{\mathbf{p} \in \mathcal{X}} \mathbf{p}^T (\bar{\mathbf{r}}_P + \gamma M_P \hat{V}^{t+1}) \tag{54}$$

Algorithm 5 can be viewed as a value iteration extension of the SC-MDP worst-case Algorithm 2. We restrict the policy set to safe policies to ensure safety constraints. The resulting decision policy maximizes the worst-case scenario.

Similar to Remark 7 after Algorithm 2, the optimal solution of (54) is not unique, and hence "argmax" is set-valued. To get a unique solution, we can select $\hat{P}$ to be the projection of the infinite-horizon unconstrained optimal MDP policy P^*_{MDP} on the set

$$\operatorname*{argmax}_{P \in \hat{\mathcal{C}}} \min_{\mathbf{p} \in \mathcal{X}} \mathbf{p}^T (\bar{\mathbf{r}}_P + \gamma M_P \hat{V}^{t+1}).$$

With this extra optimization, where $\hat{P}$ can simply be computed by a linear program analogous to the LP in Theorem 2, if the unconstrained optimal policy P^*_{MDP} is feasible (e.g., if $L = I_n$ and $\mathbf{d} = \mathbf{1}$), then $\hat{V}^* = V^*$, where V^* is the optimal value function for unconstrained MDPs given by the standard infinite-horizon value iteration, and thus, $\hat{P} = P^*_{\text{MDP}}$.

The following proposition shows that Algorithm 5 is convergent.

Proposition 3: Suppose that $0 \leq \gamma < 1$; then, the value function $\hat{V}^t$ in Algorithm 5 converges to a fixed point $\hat{V}^*$, i.e.,

$$\lim_{t \to \infty} \hat{V}^t = \hat{V}^*. \tag{55}$$

Proof: The convergence of $\hat{V}^t$ can be established by applying the standard Banach fixed-point theorem. Let $\mathcal{V}$ denote the partially ordered normed linear space of bounded real-valued functions on $\mathcal{S}$ with infinity (supremum) norm and elementwise partial ordering. For $v \in \mathcal{V}$, define the operator T on $\mathcal{V}$ by

$$Tv := \max_{P \in \hat{\mathcal{C}}} \left\{ \bar{\mathbf{r}}_P + \gamma M_P v \right\}$$

where we compute the $\max$ with respect to the elementwise partial ordering on $\mathcal{V}$. We will show first that T is a contraction mapping on $\mathcal{V}$, i.e., $||Tv - Tu||_\infty \leq \gamma ||v - u||_\infty$ for all v and u in $\mathcal{V}$. Since $\mathcal{S}$ is finite, T maps $\mathcal{V}$ into $\mathcal{V}$. Let u and v be in $\mathcal{V}$, and fix $s \in \mathcal{S}$, assume that $Tv(s) \geq Tu(s)$, and let $P^*(s) \in \operatorname*{argmax}_{P \in \hat{C}} \left\{ \bar{r}_P(s) + \gamma \sum_{j \in \mathcal{S}} M_P(s,j) v(j) \right\}$. Then,

$$\begin{aligned}
0 &\leq Tv(s) - Tu(s) \\
&\leq \bar{r}_{P^*(s)}(s) + \gamma \sum_{j \in \mathcal{S}} M_{P^*(s)}(s,j) v(j) \\
&\quad - \bar{r}_{P^*(s)}(s) - \gamma \sum_{j \in \mathcal{S}} M_{P^*(s)}(s,j) u(j) \\
&= \gamma \sum_{j \in \mathcal{S}} M_{P^*(s)}(s,j)(v(j) - u(j)) \\
&\leq \gamma \sum_{j \in \mathcal{S}} M_{P^*(s)}(s,j) ||v - u||_\infty = \gamma ||v - u||_\infty.
\end{aligned}$$

Repeating the same argument in the case that $Tu(s) \geq Tv(s)$ implies that $|Tv(s) - Tu(s)| \leq \gamma ||v - u||_\infty$ for all $s \in \mathcal{S}$, and thus, T is a contracting mapping on $\mathcal{V}$. Therefore, we can apply the Banach fixed-point theorem [11, Th. 6.2.3] to deduce that there exists a unique v^* in $\mathcal{V}$ such that $Tv^* = v^*$ and, for arbitrary v^0 in $\mathcal{V}$, the sequence $\{v^t\}$ defined by $v^{t+1} = Tv^t = T^{t+1}v^0$ converges to v^*. This procedure is the same as used in Algorithm 5, and this completes the proof. ■

Since $\hat{V}^t$ is converging, given $\epsilon > 0$ and using any vector norm $||.||$, there exists an integer $K > 0$ such that $||\hat{V}^{t+1} - \hat{V}^t|| < \epsilon$ for $t \geq K$ and the algorithm terminates in finite time.

F. Discussion

The proposed safe policy synthesis algorithms provide safety at the expense of reward. We compare in Table I the safe algorithms with the others from the literature. The table differentiates the type of constraints that the safe policies provide with the classical approaches. Unconstrained optimal MDP policies can be synthesized using either BI—as in value iteration—, policy iteration (PI), or LP. Classical constrained MDP optimal policies are synthesized using LP (cf. Section V). Safe SC-MDP policies require both a BI and a solution of LP at every step of the induction. The classical constrained and unconstrained MDP policies are optimal within their problem definitions (optimality metrics and constraints), but they can violate the hard safety constraints introduced in this paper, i.e., they may not be feasible for the SC-MDPs introduced here. The optimal safe policies for SC-MDPs are difficult to synthesize due to the nonconvexity in the objective function and the couplings between the decision variables that arise due to the hard safety constraints. The optimal safe policy is then a function of the initial state *pd*. This was the main reason for providing different safe policy synthesis algorithms with varying performance. While classical constrained MDP policies are designed for infinite-horizon problems (they satisfy constraints asymptotically), safe policies for SC-MDPs can be adapted and designed for infinite or finite horizon. Moreover, it is clear that randomization is the key for achieving feasible policies because deterministic policies can violate the safety constraints.

In Appendix B, we provide a simple two-state MDP example where we derive the analytical solution of the proposed SC-MDP

TABLE I
COMPARING DIFFERENT MDP FORMULATIONS

Problem Description	Constraint Type	Method	Solution	Horizon	Policy	Complexity
Unconstrained MDP	N/A	LP/BI/PI	Optimal	Finite/Infinite	Deterministic	Polynomial
Classical Approach	$\sum_{t=0}^{\infty}\gamma^t \mathbf{p}_t \leq \mathbf{d}$	LP	Optimal	Infinite	Randomized	Polynomial
SC-MDP	$L\mathbf{p}_t \leq \mathbf{d}$, $t = 0, 1, ...$	BI + LP	Suboptimal	Finite/Infinite	Randomized	Polynomial

algorithms and give clear insights on the contribution of this paper over the existing techniques of representing constraints.

Remark 10: The proposed algorithms to the SC-MDP problem provide a novel solution to a problem, which was not tackled before. The use of alternative methods, e.g., model checking, to solve this problem would be an interesting research direction.

VII. SIMULATIONS

In the simulations, we consider $L = I_n$ and $\mathbf{d} \in [0, 1]^n$ in (9); thus, the constraints are given by $\mathbf{p}_t \leq \mathbf{d}, t = 0, 1, \ldots$. Using the law of large numbers, these constraints can be interpreted as bounding the density of an autonomous multiagent system at every time epoch [13]. Therefore, the simulation environment would consider the application of the safety constraints to swarm control of autonomous multiagent systems.

A. Swarm Coordination Problem

This section presents an example to demonstrate the performance of the proposed methodology for SC-MPDs compared to algorithms from the literature on a vehicle swarm coordination problem [10], [13], [14]. In this application, autonomous vehicles (agents) explore a region that can be partitioned into n disjoint subregions (or bins). We can model the system as an MDP where the states of agents are their bin locations (n states) and the actions of a vehicle are defined by the possible transitions to neighboring bins. Each vehicle collects rewards while traversing the area where, due to the stochastic environment, transitions are stochastic (i.e., even if the vehicle's command is to go "North," the environment can send the vehicle with a small probability to "East"). Note that the safety constraints developed in this paper can be interpreted as follows: If the number of vehicles is large where each vehicle is moving according an identical MDP, then the density of vehicles evolves following a MC by the law of large numbers [13]. Since the physical environment (capacity/size of bins) can impose constraints on the number of vehicles in a given bin, the safety state constraints in (9) can be imposed to bound the expected number of vehicles using $L = I_n$ and $d \in [0, 1]^n$ and, hence, to mitigate conflicts. The safety state constraints are referred to as density constraints hereafter.

For simplicity, we consider the operational region to be a 3×3 grid, so there are essentially nine states. Each vehicle has the following five possible actions: go "North," "South," "East," "West," and "Stay" (see Fig. 2). When a vehicle is in a boundary bin and takes an action to go off the boundary, the environment will make it stay in the same bin. Also note that the action "Stay" is deterministic: If a vehicle decides to stay in a certain bin, the environment will not send it anywhere else.

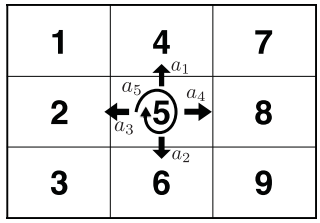


Fig. 2. Illustration of 3×3 grid describing the nine MDP states and the five actions (North, South, West, East, and Stay).

Reward $R_t(i, a)$ is assumed to be independent of the action a or time t, and only changes with the state i; thus, we can write R_t as vectors for $t = 1, \ldots, N$ and define them as follows:

$$R_t = \begin{bmatrix} 10 & 1 & 1 & 3 & 3 & 1 & 1 & 5 & 1 \end{bmatrix}^T \tag{56}$$

where $R_t(i)$ is the reward collected at bin (state) i and is assumed to be independent of the action or time. Density safety constraints for different bins are described by

$$\mathbf{d} = \begin{bmatrix} 0.6 & 1 & 1 & 0.05 & 0.05 & 1 & 1 & 1 & 1 \end{bmatrix}^T \tag{57}$$

where for each state i, $\mathbf{p}_t(i) \leq \mathbf{d}(i)$ for $t = 1, 2, \ldots$ If the density constraints are relaxed, the unconstrained MDP solution (which is known to give deterministic optimal policies) will send all the vehicles to the bin with the highest reward (in this case to bin number 1). The classical approach, on the other hand, may cause violation of constraints during transients. However, with our proposed approach, not only the constraints are satisfied at all time epochs but also the solution gives guarantees on the expected total reward. Note that the linear program (34) generates the policies independent of the initial distribution. Therefore, even if the initial distributions of vehicles were unknown (which is usually the case in autonomous swarms), the generated policy satisfies the constraints.

We now consider the case when all the vehicles initially are in bin 6, i.e.,

$$\mathbf{p}_0 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^T. \tag{58}$$

Note that this is a feasible starting density ($\mathbf{p}_0 \leq \mathbf{d}$). Fig. 3 shows the simulation with a discount factor $\gamma \simeq 1$. The unconstrained MDP policy makes the vehicles go to a single bin (bin number 1) regardless of the constraints since it gives the highest reward. The policy computed with the classical approach, on the other hand, makes the density converge to a feasible density asymptotically. However, both of these policies cause violations in bin 4 to collect higher rewards. Note that the classical approach does not prevent this transient violation. However, our policy generated from SC-MDP algorithms (Algorithms 2–5)

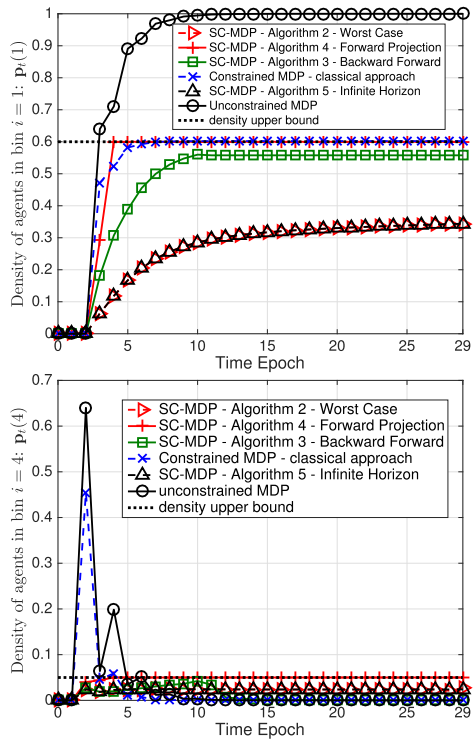


Fig. 3. Density of autonomous vehicles for each of the corresponding algorithms presented in this paper. Under the unconstrained policy, the density converges to an infeasible density distribution. The classical approach policy makes the density converge to a feasible fix point, but it causes a transient violation. This problem is resolved by the synthesized SC-MDP policies proposed in this paper.

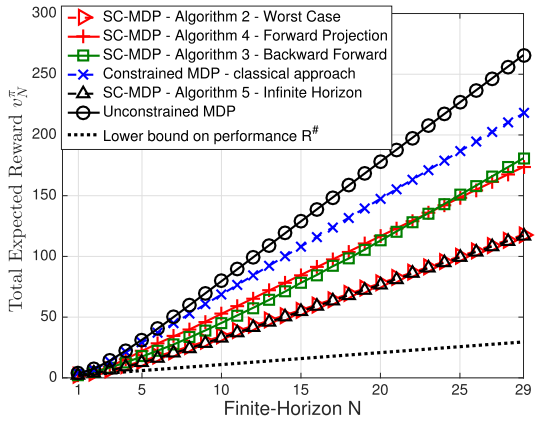


Fig. 4. Total expected rewards obtained by running the MDP policy synthesized using different algorithms. The unconstrained MDP algorithm corresponds to the optimal MDP policy without density constraints. The constrained MDP-classical algorithm corresponds to the optimal policy considering density constraints given by the classical approach. Due to the (transient) violation of the constraints, neither of them is feasible. SC-MDP feasible policies correspond to policies from the feasible set computed by the SC-MDP algorithms. The lower bound $R^{\#}$ is derived by Theorem 1.

lead to a distribution of the swarm that satisfies density constraints for all time epochs. To further investigate the efficiency of the algorithm, we have to study the rewards associated with the proposed policy. In Fig. 4, we compare the total expected reward of the SC-MDP policies with the *infeasible* policies of the unconstrained MDP and the classical approach for dealing with constraints. The lower bound $R^{\#}$ shown in the figure is derived by Theorem 1, which provides optimality guarantees for the generated policies. Fig. 4 also shows that Algorithms 2 and 5 have a similar performance; this is intuitive as Algorithm 5 is an infinite-horizon extension of Algorithm 2. Interestingly, even though Algorithms 3 and 4 are very different approaches, they provide a similar performance, which is relatively close to the performance of the, infeasible, classical approach. Note that Algorithm 3 has a further advantage over Algorithm 4 of being robust in its design to perturbations on the initial state *pd*.

VIII. Conclusion

In this paper, we studied finite-state finite-action MDP problems with hard safety constraints on the probability of being in a state of the induced MC. It is shown that neither optimal policies due to unconstrained MDP approach nor the policies generated using the classical constrained MDP approach are guaranteed to be feasible (i.e., they do not satisfy the safety constraints introduced in this paper). We provide efficient algorithms based on LP and duality theory of convex optimization to generate feasible Markovian policies that not only satisfy the constraints but also ensure some theoretical guarantees on the total expected reward. These safe policies define a *pd* over possible actions and require that agents randomize their actions as a function of the state, i.e., they are randomized policies. For a future work, it would be useful to provide tight bounds on the performance metric for the feasible policies compared with the optimal safe policies.

As a future work, it would be interesting to formulate the SC-MDP problem as a synthesis (model checking) problem and compare the solution approach developed in this paper with tools developed by the verification community. Also, characterizing the distance of the computed safe policies from the set of optimal safe policies is an interesting research direction.

Appendix A
Unconstrained MDP Optimal Policy

We provide in Algorithm 6 the *BI* algorithm [11, p. 92], also known as value iteration, based on DP that gives the optimal policy as well as the optimal value for unconstrained MDP problem.

Appendix B
Two-State MDP Example

Consider the two-state MDP shown in Fig. 5. The two-state MDP example is used to illustrate (analytically) the difference between the proposed algorithms and provides clear insights on the contribution of this paper over the existing techniques of representing constraints.

1) States $\mathcal{S} = \{s_1, s_2\}$, discount factor $\gamma = 1$.
2) Actions $\mathcal{A}_{s_1} = \{a_{1,1}, a_{1,2}\}$ and $\mathcal{A}_{s_2} = \{a_{2,1}, a_{2,2}\}$.

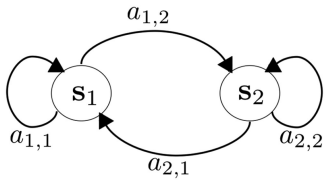


Fig. 5. Example of two-state MDP with deterministic transitions.

Algorithm 6: Backward Induction: Unconstrained MDP Optimal Policy.

1: **Definitions:** For any state $s \in \mathcal{S}$, we define $V_t^{\pi}(s) = \mathbb{E}[\sum_{k=t}^{N-1} \gamma^k R_k(X_k, A_k) + \gamma^N R_k(X_N)|X_t = s]$ and $V_t^*(s) = \max_{\pi} V_t^{\pi}$ given that $X_t = s$.
2: Start with $V_N^*(s) = \bar{r}_N(s)$
3: for $t = N-1, \ldots, 0$ given V_{t+1}^* for all $s \in \mathcal{S}$ calculate the optimal value

$$V_t^*(s) = \max_a \left\{ R_t(s,a) + \gamma \sum_{j \in \mathcal{S}} G_{t,a}(s,j) V_{t+1}^*(j) \right\}$$

and the optimal policy is defined by $P_t^*(s,a) = 1$ for $a = a_t^*(s)$, else $P_t^*(s,a) = 0$ where $a_t^*(s)$ is given by

$$a_t^*(s) = \text{argmax}_a \left\{ R_t(s,a) + \gamma \sum_{j \in \mathcal{S}} G_{t,a}(s,j) V_{t+1}^*(j) \right\}$$

4: **Result:** $v_N^* = \mathbf{p}_0^T V_0^*$.

3) Transition probabilities:

$$\begin{aligned} &\text{Prob}[s_1|s_1, a_{1,1}] = 1, \quad \text{Prob}[s_2|s_1, a_{1,1}] = 0, \\ &\text{Prob}[s_1|s_1, a_{1,2}] = 0, \quad \text{Prob}[s_2|s_1, a_{1,2}] = 1, \\ &\text{Prob}[s_2|s_2, a_{2,1}] = 0, \quad \text{Prob}[s_1|s_2, a_{2,1}] = 1, \\ &\text{Prob}[s_2|s_2, a_{2,2}] = 1, \text{ and } \quad \text{Prob}[s_1|s_2, a_{2,2}] = 0. \end{aligned}$$

4) Constant reward vector $\bar{\mathbf{r}}$: $\bar{r}_t(s_1) = 0$, $\bar{r}_t(s_2) = 1$ for all t.

1) Policy Description and Transition Matrix: Notice that, for simplicity, the rewards are selected independent of the policy, and let us consider $N = 1$, i.e., only one decision epoch policy $\pi = \{P\}$. Let $\alpha_1 = \text{Prob}[a_{1,2}|s_1]$ and $\alpha_2 = \text{Prob}[a_{2,2}|s_2]$, then the decision matrix P is given as follows:

$$P = \begin{bmatrix} 1-\alpha_1 & \alpha_1 \\ 1-\alpha_2 & \alpha_2 \end{bmatrix}.$$

The transition matrix M is then given by $M = P$.

2) Unconstrained MDP Policy: The unconstrained optimal MDP policy (see Algorithm 6) gives the following output:

$$V_1^* = \bar{\mathbf{r}} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, V_0^* = \bar{\mathbf{r}} + \begin{bmatrix} \max\{0,1\} \\ \max\{1,0\} \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix},$$
$$\alpha_1^* = 1, \alpha_2^* = 1$$
$$v^* = \mathbf{p}_0^T V_0^* = 1 + \mathbf{p}_0(s_2).$$

3) Analytical Expression of $\hat{\mathcal{C}}$ and $\hat{\mathcal{C}}_t$: In the following lemma, we will derive the analytical expression of $\hat{\mathcal{C}}$ and $\hat{\mathcal{C}}_t$ when the safety upperbound is $\mathbf{d} = \left[\begin{smallmatrix} 1 \\ 0.5 \end{smallmatrix}\right]$.

Lemma 4: $\hat{\mathcal{C}}$ and $\hat{\mathcal{C}}_t$ for the two-state example of Fig. 5 with $\mathbf{d} = \left[\begin{smallmatrix} 1 \\ 0.5 \end{smallmatrix}\right]$ are given by

$$\hat{\mathcal{C}} = \{P \in \mathbb{R}^{2,2} : \alpha_1 \leq 0.5, \alpha_1 + \alpha_2 \leq 1, \alpha_1 \geq 0, \alpha_2 \geq 0\} \tag{59}$$

and

$$\hat{\mathcal{C}}_t = \{P \in \mathbb{R}^{2,2} : \alpha_1 = 0.5, 0 \leq \alpha_2 \leq 0.5\}. \tag{60}$$

Proof: Recall that $\hat{\mathcal{C}}$ is the set of safe policies, i.e., if $P \in \hat{\mathcal{C}}$ for all t, then $\mathbf{p}_t \leq \mathbf{d}$ for all t. Using the fact that $M = P$, then P must satisfy the set of linear inequalities given in this paper for $\mathcal{M}$ (or use [10, Th. 1]). But, since the example here is simple, we will derive these equations. For P to be a safe policy, the following equations must be satisfied:

$$\left(\max_{\mathbf{p} \in \mathcal{X}} \mathbf{e}_1^T M^T \mathbf{p} \right) \leq \mathbf{d}(s_1), \text{ and } \left(\max_{\mathbf{p} \in \mathcal{X}} \mathbf{e}_2^T M^T \mathbf{p} \right) \leq \mathbf{d}(s_2)$$

where $\mathcal{X} = \{\mathbf{p} \in \mathbb{R}^2 : \mathbf{p}(s_1) + \mathbf{p}(s_2) = 1, 0 \leq \mathbf{p}(s_1) \leq 1, 0 \leq \mathbf{p}(s_2) \leq 0.5\}$, $\mathbf{e}_k$ is the vector of all zeros except for the kth entry that is equal to 1, and $M = P$. Since $\mathbf{d}(s_1) = 1$, the first equation is always satisfied for any P. Let $f(\alpha_1, \alpha_2) = \left(\max_{\mathbf{p} \in \mathcal{X}} \mathbf{e}_2^T M^T \mathbf{p}\right)$, then it can be verified that

$$f(\alpha_1, \alpha_2) = \begin{cases} \alpha_1 & \text{if } \alpha_1 \geq \alpha_2 \\ 0.5\alpha_1 + 0.5\alpha_2 & \text{if } \alpha_1 < \alpha_2. \end{cases}$$

For safety, it remains to impose $f(\alpha_1, \alpha_2) \leq 0.5$, and this can be reduced to the following equations:

$$\alpha_1 \leq 0.5 \text{ and } 0.5\alpha_1 + 0.5\alpha_2 \leq 0.5.$$

We can now give the expression of $\hat{\mathcal{C}}$

$$\hat{\mathcal{C}} = \{P \in \mathbb{R}^{2,2} : \alpha_1 \leq 0.5, \alpha_1 + \alpha_2 \leq 1, \alpha_1 \geq 0, \alpha_2 \geq 0\}. \tag{61}$$

Notice that $\left[\begin{smallmatrix} 1 & 0 \\ 1 & 0 \end{smallmatrix}\right] \in \hat{\mathcal{C}}$, but if $\mathbf{p}_0(s_1) = 1$, then the total expected reward for the MDP when $\mathbf{r}^T = [0\ 1]^T$ is

$$v^{\pi} = \sum_t \mathbf{p}_t^T \mathbf{r}_t = (\mathbf{p}_0^T + \mathbf{p}_0^T M)\mathbf{r} = 0.$$

This essentially means that having a safe policy $P \in \hat{\mathcal{C}}$ does not guarantee anything about the total expected reward. On the other hand, $\hat{\mathcal{C}}_t$ is a subset of $\hat{\mathcal{C}}$ for policies that maximize the worst

TABLE II
PERFORMANCE OF DIFFERENT MDP ALGORITHMS APPLIED TO THE TWO-STATE EXAMPLE IN FIG. 5.

Algorithm	Total Expected Reward ($\gamma = 1, N = 1$)
Lower Bound (Theorem 1)	$R^{\#} = 0.5$
Unconstrained MDP	$v_N^{\pi} = 1 + \mathbf{p}_0(s_2)$
SC-MDP Algorithm 2	$v_N^{\pi} = 0.5 + 0.5\mathbf{p}_0(s_2)$
SC-MDP Algorithm 3	$v_N^{\pi} = 0.5 + \mathbf{p}_0(s_2)$
SC-MDP Algorithm 4	$v_N^{\pi} = 0.5 + \mathbf{p}_0(s_2)$

case. Applying the equation given in Algorithm 2, we have

$$\begin{aligned}\hat{\mathcal{C}}_t &:= \underset{P\in\hat{\mathcal{C}}}{\text{argmax}} \ \min_{\mathbf{p}\in\mathcal{X}} \mathbf{p}^T U_{P_t} \\ &= \underset{P\in\hat{\mathcal{C}}}{\text{argmax}} \ \min_{\mathbf{p}\in\mathcal{X}} \mathbf{p}^T \left(\begin{bmatrix} 0 \\ 1 \end{bmatrix} + M \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right) \\ &= \underset{P\in\hat{\mathcal{C}}}{\text{argmax}} \ \min_{\mathbf{p}\in\mathcal{X}} \alpha_1 \mathbf{p}(s_1) + (1+\alpha_2)\mathbf{p}(s_2) \\ &= \underset{P\in\hat{\mathcal{C}}}{\text{argmax}} \alpha_1 .\end{aligned}$$

The last equation is because $1 + \alpha_2 > \alpha_1$, so the minimum occurs when $\mathbf{p}(s_1) = 1$ and $\mathbf{p}(s_2) = 0$. Therefore

$$\hat{\mathcal{C}}_t = \{P \in \mathbb{R}^{2,2} : \alpha_1 = 0.5, 0 \le \alpha_2 \le 0.5\}.$$

■

It can be verified from the above expression that the lower bound given in Theorem 1 is $R^{\#} = 0.5$, i.e., for all $P \in \hat{\mathcal{C}}_t$, the total expected reward is $v^{\pi} \ge 0.5$ for any initial distribution $\mathbf{p}_0$. As shown here, even with this simple example, $\hat{\mathcal{C}}_t$ is not a singleton but a set of policies that maximize the worst case. Note that Algorithm 2 only requires $0 \le \alpha_2 \le 0.5$, but by inspection the optimal policy would be $\alpha_2 = 0.5$.

4) Performance of SC-MDP Algorithms: Table II summarizes the performance of the different algorithms for the two-state example with $\mathbf{d} = \left[\begin{smallmatrix} 1 \\ 0.5 \end{smallmatrix}\right]$. We first apply Algorithm 3 with an initial distribution $\mathbf{p}_0$ that satisfies $\mathbf{p}_0(s_2) > 0$. It can be shown that starting from any policy $P^0 \in \hat{\mathcal{C}}_t$, the backward–forward converges in one iteration to

$$\begin{aligned}&\underset{P\in\hat{\mathcal{C}}_t}{\text{argmax}} \ \mathbf{p}_0^T \left(\begin{bmatrix} 0 \\ 1 \end{bmatrix} + M \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right) \\ &\quad = \underset{P\in\hat{\mathcal{C}}_t}{\text{argmax}} \ \alpha_1 \mathbf{p}_0(s_1) + (1+\alpha_2)\mathbf{p}_0(s_2).\end{aligned}$$

This gives a unique solution $\alpha_1 = 0.5$ and $\alpha_2 = 0.5$. Therefore, the solution of Algorithm 3 is the policy $P = \left[\begin{smallmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{smallmatrix}\right]$, which gives a total expected reward

$$v^{\pi} = \mathbf{p}_0(s_2) + \frac{\mathbf{p}_0(s_2) + \mathbf{p}_0(s_1)}{2} = 0.5 + \mathbf{p}_0(s_2)$$

which turns out to be the optimal safe policy. By comparing this with the optimal unconstrained MDP solution ($v^* = 1 + \mathbf{p}_0(s_2)$), we see that the total expected reward is decreased by 0.5 at the benefit of providing safety. With a similar derivation, Algorithm 4 also provides a total expected reward $v^{\pi} = 0.5 + \mathbf{p}_0(s_2)$.

Remark 11: Note that penalizing the constraint violation with a sufficiently large weighting coefficient in the cost function is not equivalent to imposing constraints explicitly. It can be easily checked that for any reward function $\bar{\mathbf{r}}$, and an initial distribution $\mathbf{p}_0$, an optimal policy is

$$\begin{cases} P^* = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}, \mathbf{p}_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} & \text{if } \bar{\mathbf{r}}(s_2) \ge \bar{\mathbf{r}}(s_1) \\ P^* = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}, \mathbf{p}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} & \text{if } \bar{\mathbf{r}}(s_2) < \bar{\mathbf{r}}(s_1). \end{cases} \tag{62}$$

Clearly, whatever the reward function was, the optimal policy would be to go to the state with the highest reward, i.e., $\mathbf{p} = \mathbf{e}_k$, where $\mathbf{e}_k$ is the vector of all zeros except for the index k corresponding to the state with the highest rewards as shown in the example above. To illustrate, consider the safety constraint $\mathbf{d} = \left[\begin{smallmatrix} 0.9 \\ 0.6 \end{smallmatrix}\right]$, which bounds the probability of being in states s_1 and s_2, i.e., $\text{Prob}[X_t = s_2] \le 0.6$ and $\text{Prob}[X_t = s_1] \le 0.9$. The optimal MDP policy for the unconstrained case with any weighted reward function is not a feasible solution for SC-MDP because the optimal policy leads to a distribution with a state s having $\mathbf{p}_1(s) > \mathbf{d}(s)$.

Remark 12: For the general case, the ordering of the performance of the algorithms is given by

$$R^{\#} \le v_N^{\pi_{\text{Algorithm 2}}} \le v_N^{\pi_{\text{Algorithm 3}}} \le v_N^{\pi_{\text{Algorithm 6}}}$$

where $v_N^{\pi_{\text{Algorithm 2}}}$ is the performance of the worst-case policy that Algorithm 2 can generate, and

$$v_N^{\pi_{\text{Algorithm 4}}} \le v_N^{\pi_{\text{Algorithm 6}}}.$$

Therefore, Algorithm 4 performance in comparison to other algorithms depends on the specific problem at hand.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their helpful and constructive comments that greatly contributed to improving the final version of this paper. We would also like to thank J. Fu of Worcester Polytechnic Institute for her feedback on the connection between our proposed approach and formal methods. The research in this paper was carried out in part at the Jet Propulsion Laboratory of the National Aeronautics and Space Administration (JPL-NASA). This paper extends the conference paper [1] by generalizing the theorems, providing additional theoretical results (Propositions 1 and 2), adding more algorithms (Algorithms 3 and 4) that enhance the performance of safe MDP policies, and extending the algorithms to infinite-horizon MDPs (Algorithm 5).

REFERENCES

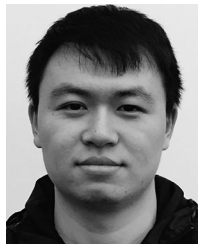
[1] M. El Chamie, Y. Yu, and B. Açıkmeşe, "Convex synthesis of randomized policies for controlled Markov chains with density safety upper bound constraints," in *Proc. Amer. Control Conf.*, Jul. 2016, pp. 6290–6295.

[2] E. Feinberg and A. Shwartz, *Handbook of Markov Decision Processes: Methods and Applications* (International Series in Operations Research & Management Science). New York, NY, USA: Springer-Verlag, 2002.

[3] E. Altman, "Applications of Markov decision processes in communication networks: A Survey," French Inst. Res. Comput. Sci. Autom., Rocquencourt, France, Rep. RR-3984, 2000.
[4] N. Demirer, M. El Chamie, and B. Açıkmeşe, "Safe Markov chains for ON/OFF density control with observed transitions," *IEEE Trans. Autom. Control*, vol. 63, no. 5, pp. 1442–1449, May 2018.
[5] R. Zhang, Y. Yu, M. El Chamie, B. Açıkmeşe, and D. Ballard, "Decision-making policies for heterogeneous autonomous multi-agent systems with safety constraints," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, 2016, pp. 546–552.
[6] V. Krishnan and S. Martínez, "Distributed control for spatial self-organization of multi-agent swarms," 2017. [Online]. Available: arXiv:1705.03109
[7] S. Martínez, J. Cortes, and F. Bullo, "Motion coordination with distributed information," *IEEE Control Syst.*, vol. 27, no. 4, pp. 75–88, Aug. 2007.
[8] A. Arapostathis, R. Kumar, and S. Tangirala, "Controlled Markov chains with safety upper bound," *IEEE Trans. Autom. Control*, vol. 48, no. 7, pp. 1230–1234, Jul. 2003.
[9] S. Hsu, A. Arapostathis, and R. Kumar, "On optimal control of Markov chains with safety constraint," in *Proc. Amer. Control Conf.*, 2006, pp. 4516–4521.
[10] B. Açıkmeşe, N. Demir, and M. Harris, "Convex necessary and sufficient conditions for density safety constraints in Markov chain synthesis," *IEEE Trans. Autom. Control*, vol. 60, no. 10, pp. 2813–2818, Oct. 2015.
[11] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming* (Wiley Series in Probability and Mathematical Statistics). New York, NY, USA: Wiley, 1994.
[12] E. Altman, *Constrained Markov Decision Processes* (Stochastic Modeling Series). New York, NY, USA: Taylor & Francis, 1999.
[13] N. Demir, U. Eren, and B. Açıkmeşe, "Decentralized probabilistic density control of autonomous swarms with safety constraints," *Auton. Robots*, vol. 39, no. 4, pp. 537–554, 2015.
[14] B. Açıkmeşe and D. S. Bayard, "Markov chain approach to probabilistic guidance for swarms of autonomous agents," *Asian J. Control*, vol. 17, no. 4, pp. 1105–1124, 2015.
[15] E. A. Feinberg and A. Shwartz, "Constrained Markov decision models with weighted discounted rewards," *Math. Oper. Res.*, vol. 20, no. 2, pp. 302–320, 1995.
[16] M. Haviv, "On constrained Markov decision processes," *Oper. Res. Lett.*, vol. 19, no. 1, pp. 25–28, 1996.
[17] E. Altman and A. Shwartz, "Markov decision problems and state-action frequencies," *SIAM J. Control Optim.*, vol. 29, no. 4, pp. 786–809, Jul. 1991.
[18] A. Arapostathis, R. Kumar, and S. P. Hsu, "Control of Markov chains with safety bounds," *IEEE Trans. Autom. Sci. Eng.*, vol. 2, no. 4, pp. 333–343, Oct. 2005.
[19] M. Lahijanian, S. B. Andersson, and C. Belta, "Temporal logic motion planning and control with probabilistic satisfaction guarantees," *IEEE Trans. Robot.*, vol. 28, no. 2, pp. 396–409, Apr. 2012.
[20] X. Ding, S. L. Smith, C. Belta, and D. Rus, "Optimal control of Markov decision processes with linear temporal logic constraints," *IEEE Trans. Autom. Control*, vol. 59, no. 5, pp. 1244–1257, May 2014.
[21] E. M. Wolff, U. Topcu, and R. M. Murray, "Robust control of uncertain Markov decision processes with temporal logic specifications," in *Proc. 51st IEEE Conf. Decision Control*, Dec. 2012, pp. 3372–3379.
[22] S. Coogan, M. Arcak, and C. Belta, "Formal methods for control of traffic flow: Automated control synthesis from finite-state transition models," *IEEE Control Syst.*, vol. 37, no. 2, pp. 109–128, Apr. 2017.
[23] A. Abate, J.-P. Katoen, J. Lygeros, and M. Prandini, "Approximate model checking of stochastic hybrid systems," *Eur. J. Control*, vol. 16, no. 6, pp. 624–641, 2010.
[24] C. Baier and J.-P. Katoen, *Principles of Model Checking* (Representation and Mind Series). Cambridge, MA, USA: MIT Press, 2008.
[25] M. Kwiatkowska, G. Norman, and D. Parker, "Probabilistic model checking in practice: Case studies with prism," *SIGMETRICS Perform. Eval. Rev.*, vol. 32, no. 4, pp. 16–21, Mar. 2005.
[26] S. Feyzabadi and S. Carpin, "Risk-aware path planning using hirerachical constrained Markov decision processes," in *Proc. IEEE Int. Conf. Autom. Sci. Eng.*, Aug. 2014, pp. 297–303.
[27] S. Balachandran and E. M. Atkins, "A constrained Markov decision process for flight safety assessment and management," in *Proc. AIAA Infotech @ Aerosp. Conf., AIAA SciTech Forum* Kissimmee, FL, USA, Jan. 5–9, 2015, doi: 10.2514/6.2015-0115.
[28] M. Ono, Y. Kuwata, and J. Balaram, "Mixed-strategy chance constrained optimal control," in *Proc. Amer. Control Conf.*, Jun. 2013, pp. 4666–4673.
[29] M. Ono, M. Pavone, Y. Kuwata, and J. Balaram, "Chance-constrained dynamic programming with application to risk-aware robotic space exploration," *Auton. Robots*, vol. 39, no. 4, pp. 555–571, 2015.
[30] M. El Chamie and G. Neglia, "Newton's method for constrained norm minimization and its application to weighted graph problems," in *Proc. Amer. Control Conf.*, Jun. 2014, pp. 2983–2988.
[31] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, 3rd ed. Belmont, MA, USA: Athena Scientific, 2005, vol. I.

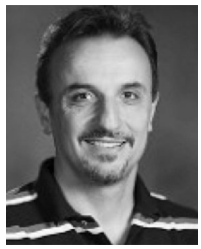
Mahmoud El Chamie received the M.S. (UBINET program) and Ph.D. degrees in computer science from the University of Nice Sophia Antipolis, Nice, France, in 2011 and 2014, respectively. He conducted the Ph.D. research at Inria Sophia Antipolis, Valbonne, France.

He is currently a Senior Research Engineer with the United Technologies Research Center (UTRC), East Hartford, CT, USA. Prior to joining UTRC, he held Postdoctoral positions at the University of Washington, Seattle, WA, and the University of Texas at Austin, and held a Visiting Postdoctoral position at NASA's Jet Propulsion Laboratory. His research interests include optimization and control of dynamical systems and their applications to robotics, aerospace, and machine intelligence.



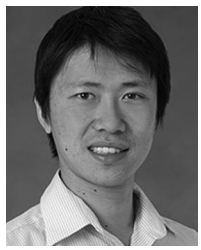
Yue Yu received the B.S. degree in aeronautics and astronautics from Beihang University, Beijing, China, in 2014. He is currently working toward the Ph.D. degree in aeronautics and astronautics from the University of Washington, Seattle, WA, USA.

His research interests include optimization, control theory, and multiagent systems.



Behçet Açıkmeşe received the Ph.D. degree from Purdue University, West Lafayette, IN, USA, in 2002.

He was a Visiting Assistant Professor with Purdue University. In 2003, he joined NASA's Jet Propulsion Laboratory as a Senior Technologist. He was a Lecturer with the California Institute of Technology, where he developed GN&C algorithms for planetary landing, formation flying spacecraft, and asteroid and comet sample return missions. From 2012 to 2015, he was a Faculty with the University of Texas at Austin. He is the Developer of the "flyaway" GN&C algorithms in Mars Science Laboratory, which landed on Mars in August, 2012. He is currently an Associate Professor with the Department of Aeronautics and Astronautics at the University of Washington, Seattle, WA, USA.



Masahiro Ono received the B.S. degree in aeronautics and astronautics from the University of Tokyo, Tokyo, Japan, in 2005, and the M.S. and Ph.D. degrees in aeronautics and astronautics as well as the M.S. degree in technology and policy from the Massachusetts Institute of Technology, Cambridge, MA, USA, in 2007, 2012, and 2012, respectively.

He was an Assistant Professor with Keio University. In 2013, he joined NASA's Jet Propulsion Laboratory, where he is currently a Research Technologist in the Robotic Controls and Estimation Group. His research interests include risk sensitive planning/control that enables unmanned probes to reliably operate in highly uncertain environments, such as planetary surface. His technical skills include optimization, path planning, robust and optimal control, state estimation, and automated planning and scheduling.