

Verification of Markov Decision Processes with Risk-Sensitive Measures

Murat Cubuktepe and Ufuk Topcu

Abstract—We develop a method for computing policies in Markov decision processes with risk-sensitive measures subject to temporal logic constraints. Specifically, we use a particular risk-sensitive measure from cumulative prospect theory, which has been previously adopted in psychology and economics. The nonlinear transformation of the probabilities and utility functions yields a nonlinear programming problem, which makes computation of optimal policies typically challenging. We show that this nonlinear weighting function can be accurately approximated by the difference of two convex functions. This observation enables efficient policy computation using convex-concave programming. We demonstrate the effectiveness of the approach on several scenarios.

I. INTRODUCTION

Markov decision processes (MDPs) model sequential decision-making problems in stochastic dynamic environments [30]. MDP formulations typically focus on the *risk-neutral* expected cost or reward model. On the other hand, MDPs with *risk-sensitive* measures, such as exponential utility [17], percentile risk criteria [14] and conditional value at risk [13], [12], [34] have been studied in the literature have also found applications in portfolio management [8], robotics [27], stochastic shortest-path problems [7], optimal control [16] and operations research [10], [17]. These measures capture the variability in the cost due to stochastic transitions in an MDP, and aim to minimize the effect of the outcomes with high cost.

We focus on a particular risk-sensitive measure that comes from *cumulative prospect theory* (CPT) [35]. This measure is widely used in psychology and economics to build models that explain the risk-sensitive behavior of humans in decision-making. Empirical evidence suggest CPT characterizes human preferences in decision-making [24], [35]. The key elements of this theory are a value function that is concave for gains, convex for losses, and steeper for losses than for gains, and a *nonlinear* transformation of the probability range, which inflates small probabilities and deflates high probabilities. It is also a generalization of other risk-sensitive measures like VaR or CVaR [28]. Additionally, with different nonlinear weighting functions, CPT-based measures can represent risk-taking measures as well as risk-averse measures.

We investigate *model checking* with respect to *temporal logic specifications*. *Formal verification* of temporal logic specifications has been extensively studied for MDPs with risk-neutral measures [5], and mature tools exist for efficient

verification with such risk-neutral measures [19]. Probabilistic model checking verifies reachability properties such as “the probability of reaching a set of unsafe states is less than 5%” and expected costs properties such as “the expected cost of reaching a goal state is less than 10%”. A rich set of properties, specified by temporal logic specifications, can be reduced to *reachability* properties, which can then be verified automatically [18]. To the best of our knowledge, formal quantitative verification with respect to risk-sensitive measures has not been considered in the literature.

Dynamic programming equations for MDPs with CPT-based measures for finite-horizon MDPs in [22] and for infinite-horizon MDPs in [21] exist. However, computing an optimal policy requires optimizing integrals of nonlinear functions over continuous variables, which can be computationally impractical. CPT-based measures have been used in reinforcement learning [28], where it was shown that the policy gradient approach converges to the optimal CPT value asymptotically.

The main challenge in computing policies with CPT-based measures is the nonlinear transformation of the probability range and utilities. This transformation yields a nonlinear programming problem. For efficient verification of MDPs with CPT-based measures, we approximate the nonlinear CPT weighting function by a *difference of convex* function to utilize *convex-concave procedure* [23], which efficiently computes locally optimal solutions for optimization problems with difference of convex functions. We propose methods to approximate the CPT weighting function, and discuss the trade-offs between different approximations. Experimental results show the applicability of our approach in numerical experiments.

II. PRELIMINARIES

Definition 1 (Distribution): A *probability distribution* over a finite or countably infinite set X is a function $\mu: X \rightarrow [0, 1] \subseteq \mathbb{R}$ with $\sum_{x \in X} \mu(x) = 1$. The set of all distributions on X is denoted by $\text{Distr}(X)$.

Definition 2 (Monomials, Posynomials): Let $V = \{x_1, \dots, x_n\}$ be a finite set of strictly positive real-valued variables. A *monomial* over V is an expression of the form

$$g = c \cdot x_1^{a_1} \cdots x_n^{a_n},$$

where $c \in \mathbb{R}$ is a real coefficient, and $a_i \in \mathbb{R}$ are exponents for $1 \leq i \leq n$. A *posynomial* over V is a sum of one or more monomials:

$$f = \sum_{k=1}^K c_k \cdot x_1^{a_{1k}} \cdots x_n^{a_{nk}}. \quad (1)$$

Authors are with the Department of Aerospace Engineering and Engineering Mechanics, University of Texas, 201 E 24th St, Austin, TX 78712, USA. e-mail: {mcubuktepe, utopcu}@utexas.edu. The work has been supported partly by DARPA W911NF-16-1-0001, AFRL FA8650-15-C-2546 and ONR N00014-15-IP-00052.

Definition 3 (Markov decision process): A Markov decision process (MDP) is a tuple $\mathcal{M} = (S, s_I, Act, V, \mathcal{P})$ with a finite set S of states, an initial state $s_I \in S$, a finite set Act of actions, and a transition function $\mathcal{P}: S \times Act \times S \rightarrow Distr(S)$ satisfying for all $s \in S$: $Act(s) \neq \emptyset$, where $Act(s) = \{\alpha \in Act \mid \exists s' \in S. \mathcal{P}(s, \alpha, s') \neq 0\}$. For given a state s , we denote the set of successor states by $S(s)$. A state s' is in $S(s)$ if there exists an $\alpha \in Act$ such that $\mathcal{P}(s, \alpha, s') > 0$. If for all $s \in S$ it holds that $|Act(s)| = 1$, $\mathcal{M}$ is called a discrete-time Markov chain (MC).

$Act(s)$ is the set of enabled actions at state s ; as $Act(s) \neq \emptyset$, there are no deadlock states. Costs are defined using a state-action cost function $C_t: S \times Act \times T \rightarrow \mathbb{R}_+$. Rewards are defined similarly.

Definition 4 (Policy): Given a finite horizon T , a (randomized) policy for an MDP $\mathcal{M}$ is a function $\sigma: S \times T \rightarrow Distr(Act)$ such that $\sigma(s_t, \alpha) > 0$ implies $\alpha \in Act(s_t)$ at time t . The set of memoryless policies over $\mathcal{M}$ at time t is denoted by $Pol_t^{\mathcal{M}}$, which only depends on the current state.

Definition 5 (Induced Markov chain): For MDP $\mathcal{M} = (S, s_I, Act, \mathcal{P})$ and policy $\sigma \in Pol^{\mathcal{M}}$, the Markov chain induced by $\mathcal{M}$ and σ is $\mathcal{M}^\sigma = (S, s_I, Act, \mathcal{P}^\sigma)$ where for all $s, s' \in S$,

$$\mathcal{P}^\sigma(s, s') = \sum_{\alpha \in Act(s)} \sigma(s)(\alpha) \cdot \mathcal{P}(s, \alpha, s').$$

We consider reachability properties. For Markov chain $\mathcal{D}$ with states S , let $\Pr_s^{\mathcal{D}}(\diamond T)$ denote the probability of reaching a set $T \subseteq S$ of target states from state $s \in S$; simply, $\Pr^{\mathcal{D}}(\diamond T)$ denotes the probability for initial state s_I . We use the standard probability measure as in [5]. The interest of this paper is a synthesis problem, where the objective is to find a policy in $Pol_t^{\mathcal{M}}$ such that the probability $\Pr^{\mathcal{D}}(\diamond T)$ of satisfying the reachability property is maximized or minimized.

The classical risk-neutral MDP problem is [30]

$$\inf_{\pi \in Pol^{\mathcal{M}}} \mathbb{E} \left[\sum_{t=0}^T C_t(s_t, a_t) \right]. \quad (2)$$

The problem in (2) can be solved with value iteration, policy iteration or linear programming, and the optimal policy will be a deterministic memoryless policy. The optimal policy for problem (2) will maximize the probability of satisfying the reachability property or minimize the expected cost, therefore it is a risk-neutral solution. Following from [32], we consider the risk-sensitive value function starting at s_0 , with a policy $Pol^{\mathcal{M}}$, and the resulting trajectory $(s_0, Pol_0^{\mathcal{M}}, s_1, Pol_1^{\mathcal{M}}, \dots, s_T)$, which is given by $C_T(Pol^{\mathcal{M}}, s_0) = \rho_0(c(s_0, Pol_0^{\mathcal{M}}) + \rho_1(c(s_1, Pol_1^{\mathcal{M}}) + \dots + \rho_{T-1}(c(s_{T-1}, Pol_{T-1}^{\mathcal{M}}) + C_T(s_T)) \dots))$, where ρ_t is a one-step conditional risk measure at time t . Then, we consider the following optimization problem where ρ is replaced by a CPT-based measure:

$$\inf_{\pi \in Pol^{\mathcal{M}}} C_T(Pol^{\mathcal{M}}, s_I). \quad (3)$$

A dynamic programming equation exists for the problem in (3), and the optimal policies are memoryless [32]. Any

CPT-based measure is a one-step conditional risk measure, therefore the problem (3) can be solved by solving the dynamic programming equations [22].

III. CUMULATIVE PROSPECT THEORY (CPT)

For a random variable X , the CPT value is a generalization of the expected value of X with a utility function that is concave on gains and convex on losses, and a probability weighting function that transforms the probability measure such that it inflates small probabilities and reduces larger probabilities.

Definition 6 (CPT value): For a random variable X , the CPT value is defined as

$$C(X) = \int_0^\infty w_+(\mathbb{P}(u_+(X) > z)) dz - \int_0^\infty w_-(\mathbb{P}(u_-(X) > z)) dz, \quad (4)$$

where w_+ and $w_-: [0, 1] \rightarrow [0, 1]$ are two continuous non-decreasing functions with $w_+(0) = w_-(0) = 0$ and $w_+(1) = w_-(1) = 1$, u_+ and $u_-: \mathbb{R} \rightarrow \mathbb{R}_+$ are two utility functions.

Remark 1: CPT value generalizes the expected value of a random variable, i.e., $C(X) = \mathbb{E}[X] = \int_0^\infty (\mathbb{P}(X > z)) dz - \int_0^\infty (\mathbb{P}(-X > z)) dz$, when $u_+(x) = u_-(x) = x$, and $w_+(x) = w_-(x) = x$.

The functions w_+ and w_- are the weighting functions that capture the concept of humans deflating high probabilities and inflating low probabilities when they make decisions under uncertainty. For instance, consider a scenario where one can earn \$100 with probability 1/100 and nothing otherwise, or can earn \$1 with probability 1. It is shown that the humans tend to choose the former option [35], [6], showing that the value of a decision by a human is nonlinear with respect to the transition probabilities. Reference [29] suggests the weighting function $w(k) = \exp(-0.5(-\ln k)^\eta)$, with $0 < \eta < 1$ and [35] suggests $w(k) = \frac{k^\eta}{(k^\eta + (1-k)^\eta)^{1/\eta}}$.

Both of the functions have a similar inverted-S shape and they are concave for small values of p , and convex for large values of p .

The utility functions u_+ and u_- represent how humans value gains ($X \geq 0$) and losses ($X \leq 0$) separately. For example, if we change the scenario in the above paragraph into losses, i.e., one will lose \$100 with probability 1/100 and nothing otherwise, or will lose \$1 with probability 1, then the humans tend to choose the latter option, showing that there is a difference between evaluating the gains and losses, and the CPT-based measures can handle losses and gains separately. A suggestion for the utility function is given in [35], which is $u_+(x) = |x|^m$, and $u_-(x) = -2.25|x|^m$, with $m = 0.88$. Note that, u_+ is a concave function for $x > 0$, and u_- is a convex function for $x < 0$.

Remark 2 ([28]): CPT-based measures generalize other risk-sensitive measures. For example, it is possible to represent value at risk or conditional value at risk by proper choice of weighting and utility functions.

IV. MDPs WITH CPT-BASED MEASURES

Reference [22] shows the existence of a dynamic programming equation in an MDP with CPT-based measures, and the optimal policy that comes from the dynamic programming equation is a memoryless randomized policy. Dynamic programming equations can be solved as a nonlinear programming problem. Specifically, the objective is a nonlinear function and the objective is minimized or maximized over randomized policies for a given state and time. However, solving optimization problems with a nonlinear objective function is generally impractical [22].

To come up with a scalable procedure, we approximate the weighting function by a function that is the difference of two convex functions, which will reformulate the nonlinear programming problem to a difference of convex problem. Methods such as branch and bound methods [20] or cutting plane methods [3] can find the globally optimal solution for a difference of convex problem, but these methods can be slow in practice. Instead of seeking a global solution, a locally optimal (approximate) solution can be found by utilizing the techniques of general nonlinear optimization [26].

Definition 7 (Difference of convex problem): Difference of convex (DC) problems have the following form

$$\begin{aligned} & \text{minimize} && f_0(x) - g_0(x) \\ & \text{subject to} && f_i(x) - g_i(x) \leq 0, \quad i = 1, 2, \dots, m, \end{aligned}$$

where $x \in \mathbb{R}^n$ is the variable vector, and the functions $f_i, g_i : \mathbb{R}^n \rightarrow \mathbb{R}$ for $i = 0, 1, \dots, m$ are convex.

The convex-concave procedure (CCP) [23], [33] is a heuristic algorithm for finding a locally optimal solution to a DC problem. As a first step, we replace concave functions with a convex upper bound. We then solve the approximate convex problem, and the optimal value of the approximate problem will be an upper bound of the original problem at each iteration. The CCP algorithm to solve DC problems is described in Algorithm 1.

Algorithm 1: CCP algorithm

given an initial feasible point x_0 and convex functions f_i, g_i .
 $k=0$
repeat
 1. *Convexify.* $\hat{g}_i(x) = g_i(x_k) + \nabla g_i(x_k)^T(x - x_k)$
 for $i=0,1,\dots,m$
 2. *Solve.* Set the value of x_{k+1} to the solution of the convex problem
 minimize $f_0(x) - \hat{g}_0(x)$
 subject to $f_i(x) - \hat{g}_i(x) \leq 0, \quad \text{for } i=1,2,\dots,m$.
 3. *Update iteration.* $k = k + 1$,
until stopping criterion is satisfied.

Given an initial feasible point for a DC problem, (e.g. any policy from $\text{Distr}(\text{Act})$), all of the successive iterates in Algorithm 1 will be feasible. The procedure given by Algorithm 1 is a descent algorithm, i.e, the objective will monotonically decrease over the iterations for a minimization problem or it will increase for a maximization problem, and it will converge to a local optimum [23]. Therefore, the above

algorithm can be used to compute locally optimal solutions by solving a sequence of convex optimization problems, which is efficiently solvable by well-studied methods [11].

A. Approximating the weighting function with a DC function

In general, CPT weighting functions are nonlinear functions, and we can not use the weighting functions directly in convex-concave programming. Therefore, we approximate the weighting functions by a DC function to utilize convex-concave programming. A possible way to approximate the weighting function is least-squares polynomial approximation [9] or Chebyshev polynomial approximation [25], but these methods can be inaccurate, as the CPT weighting functions that are frequently used in the literature are not Lipschitz continuous around zero probability. See Figure 1 for an example where the Chebyshev approximation method fails to approximate a weighting function.

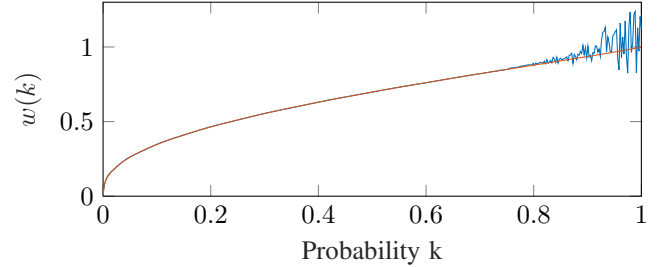


Fig. 1: An example of a CPT weighting function (red) and approximation of the CPT weighting function (blue) by a 25th degree Chebyshev polynomial with error tolerance $\epsilon = 10^{-4}$. As the CPT weighting function is not Lipschitz, the approximation with a Chebyshev basis diverges with smaller ϵ with larger values of p .

Since the Chebyshev and least-squares polynomial methods perform poorly, we modify the least-squares polynomial approximation method by extending the polynomial basis functions with monomial basis functions to accurately approximate the CPT weighting function. For example, we approximate the function $\exp(-0.5(-\ln(k))^{0.9})$, which is used in [22], by a posynomial function, $0.00231k^{0.05} + 0.00128k^{0.1} + 0.19578k^{0.35} + 0.59897k^{0.4} + 0.15968k^{0.95} + 0.03318k^3 + 0.00847k^{23}$. Figure 2 shows the posynomial function and the CPT weighting function.

B. Computing locally optimal policies

When the weighting functions are given as $w_+(x) = w_-(x) = x$ and similarly for utility functions $u_+(x) = u_-(x) = x$, the dynamic programming equation to find the policy that maximizes the probability of satisfying the reachability property is

$$\begin{aligned} p_t(s) &= \max_{\alpha \in \text{Act}} \sum_{s' \in S} \mathcal{P}(s, \alpha, s') \cdot p_{t+1}(s'), \\ p_T(s) &= 1, \quad \forall s \in Q, \quad t = 1, \dots, T, \end{aligned} \quad (5)$$

where $p_t(s)$ denotes the probability of satisfying the reachability property at state s and time t . Equivalently, we can write the dynamic programming equation in following for a given state s and time t :

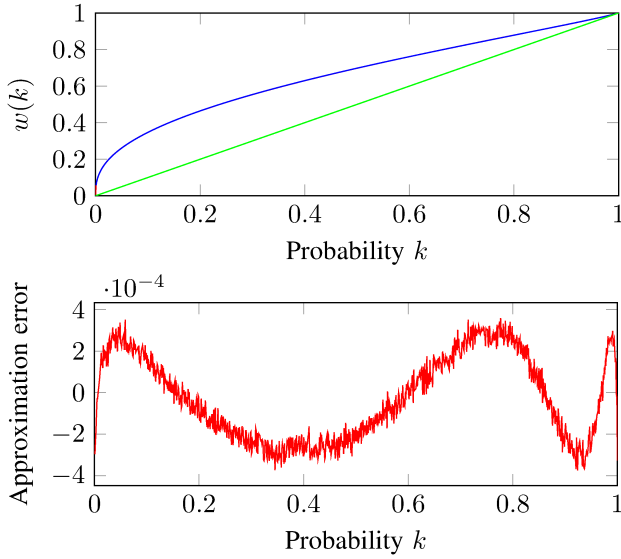


Fig. 2: Top: An example of a CPT weighting function (blue) versus a regular transition function (green) and approximation of the CPT weighting function (red) by a DC posynomial function. Note that curve of the approximation is not visible, which shows that the approximation is accurate. Bottom: The error of the approximation of the CPT weighting function by a DC posynomial function.

$$\begin{aligned}
 & \text{maximize} \quad \sum_{\sigma(s,a), a \in \text{Act}} \sum_{\alpha \in \text{Act}(s)} \sum_{s' \in S} \sigma(s, \alpha) \cdot \mathcal{P}(s, \alpha, s') \cdot p_{t+1}(s') \\
 & \text{subject to} \quad \sum_{\alpha \in \text{Act}(s)} \sigma(s, \alpha) = 1, \quad \forall \alpha \in \text{Act}(s), \quad \sigma(s, \alpha) \geq 0. \quad (6)
 \end{aligned}$$

The optimization problem in (6) maximizes the expected value of the probability for satisfying the reachability property, therefore it is a risk-neutral solution. Note that we can compute the expected value by solving the following problem:

$$\begin{aligned}
 & \text{maximize} \quad \sum_{\sigma(s,a), a \in \text{Act}} \sum_{s'_q \in S(s)} \left(\Phi_q \cdot (p_{t+1}(s'_q) - p_{t+1}(s'_{q-1})) \right) \\
 & \text{subject to} \quad \sum_{\alpha \in \text{Act}(s)} \sigma(s, \alpha) = 1, \quad \forall \alpha \in \text{Act}(s), \quad \sigma(s, \alpha) \geq 0, \\
 & \Phi_q = \sum_{\alpha \in \text{Act}(s)} \sum_{s'_m \in S(s)} \sigma(s, \alpha) \cdot \mathcal{P}(s, \alpha, s'_m), \quad (7)
 \end{aligned}$$

where $q = 1, 2, \dots, |S(s)|$ gives the index of the state in $S(s)$ after it is sorted with increasing probability of satisfying the property at time $t+1$, i.e., they are sorted with the values $p_{t+1}(s'_q)$, and $p_{t+1}(s'_0) = 0$.

The sum of the objective in (7) is over the successor states. The first sum in Φ_q is over the actions, and the second sum in Φ_q computes the probability of transitioning the successor state with at least probability $p_{t+1}(s'_{q-1})$ as a function the policy.

The problem in (7) can be viewed as maximizing the Riemann integral of the expected value, and the problem in (6) maximizes the Lebesgue integral. See the Figure 3 as an example from the MDP in Figure 4 with $\sigma(1, a) = 0.3$ and $\sigma(1, a) = 0.7$. Both problems will maximize the expected value, i.e., the area under the curve in Figure 3.

Note that the probability of satisfying the specification up to 0.2 probability is 1, regardless of the policy we choose, as 0.2 is the lowest probability of the successor states. Then, the probability of transitioning a state with at least 0.5 probability of satisfying the property can be obtained by the sum of the probabilities of transitioning the state 3 and state 4, which is given by $\sigma(s, a) + 0.4 \cdot \sigma(s, b)$ in the MDP in Example 1. Similarly, the probability of transitioning to state 4 is $0.4 \cdot \sigma(s, b)$, which gives the probability of satisfying the specification with 0.9 probability.

When $w_+(x) = w_-(x) = x$ and $u_+(x) = u_-(x) = x$, both problems in (6) and (7) can be used to maximize the expected value of satisfying the property. However, with general weighting and utility functions, we cannot use the formulation in (6), as $w(x+y) \neq w(x) + w(y)$ in general. Therefore, with a CPT weighting and utility function, we use a modified version of (7), because we can approximate the weighting function accurately.

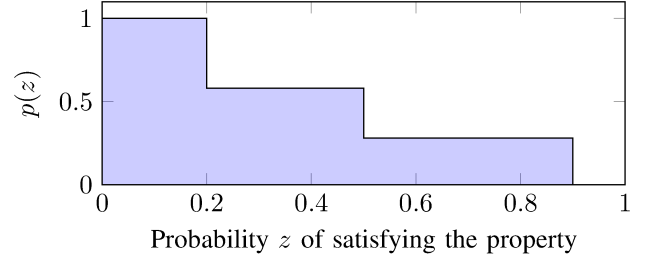


Fig. 3: The graph of the random variable with respect to the probability of satisfying the property versus probability of obtaining that value in Example 1.

Example 1: Consider the MDP in Figure 4 with 4 states at time t with $p_{t+1}(2) = 0.2$, $p_{t+1}(3) = 0.5$, $p_{t+1}(4) = 0.9$. The linear program that computes the maximum probability of satisfying the specification is:

$$\begin{aligned}
 & \text{maximize} \quad \left((\sigma(s, a) + \sigma(s, b)) \cdot (0.2) + \right. \\
 & \quad \left. (\sigma(s, a) + 0.4 \cdot \sigma(s, b)) \cdot (0.5 - 0.2) + \right. \\
 & \quad \left. (0.4 \cdot \sigma(s, b)) \cdot (0.9 - 0.5) \right)
 \end{aligned}$$

subject to $\sigma(s, a) + \sigma(s, b) = 1, \sigma(s, a) \geq 0, \sigma(s, b) \geq 0$.

For general CPT weighting and utility functions, we approximate the CPT weighting function by a posynomial. Then, for a given state s and horizon t and the approximation function $f(p)$ with K monomials, we compute a locally optimal policy by solving the following problem:

$$\begin{aligned}
 & \text{maximize} \quad \sum_{\sigma(s,a), a \in \text{Act}} \sum_{s'_q \in S(s)} \left(\Phi'_q \cdot (u_+(p_{t+1}(s'_q)) - u_+(p_{t+1}(s'_{q-1}))) \right)
 \end{aligned}$$

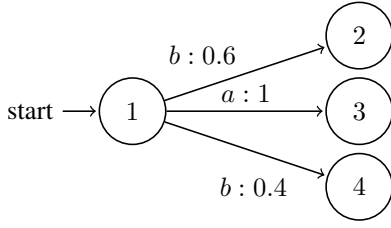


Fig. 4: An MDP with 4 states. The label $a : \gamma$ on the transitions represents that the transition happens with probability γ , when the a action is taken.

subject to

$$\begin{aligned} \sum_{\alpha \in Act(s)} \sigma(s, \alpha) &= 1, \quad \forall \alpha \in Act(s), \quad \sigma(s, \alpha) \geq 0, \\ \Phi'_q &= \sum_{k=1}^K c_k \cdot \left(\sum_{\alpha \in Act(s)} \sum_{s'_m \in S(s)} \sigma(s, \alpha) \cdot \mathcal{P}(s, \alpha, s'_m) \right)^{a_k}. \end{aligned} \quad (8)$$

We highlight the differences between the optimization problems in (7) and (8). First difference is, we replace Φ_q in (7) to Φ'_q in (8). Φ_q computes the expected value of the probability of transitioning to another state with the successor state with at least probability $p_{t+1}(s'_{q-1})$ as a function of the policy, and it is used in the risk-neutral measure. On the other hand, Φ'_q approximately computes the expected value of the probability with respect to the CPT weighting function in a CPT-based measure by approximating the CPT weighting function with $f(p)$.

The second difference is, $u_+(x) = x$ in the problem (7). Therefore, we replace $p_{t+1}(s'_q)$ to $u_+(p_{t+1}(s'_q))$. Note that the probability of satisfying the property is always between 0 and 1, therefore we use u_+ in the objective in problem (8).

Let $\beta \in \mathbb{R}^+$ and $g : \mathbb{R}^+ \rightarrow \mathbb{R}^+$. The function $g(y) = y^\beta$ is concave for $0 \leq \beta \leq 1$, and convex for $\beta \geq 1$ [11]. Therefore the problem in (8) is a DC problem, and Algorithm 1 computes a locally optimal policy.

Remark 3: For rational p , the function y^p can be represented by linear matrix inequalities (LMIs) [1]. For instance, the constraints $y^3 \leq x$ and $y \geq 0$ are equivalent to

$$\begin{bmatrix} z & y \\ y & 1 \end{bmatrix} \geq 0 \quad \text{and} \quad \begin{bmatrix} x & z \\ z & y \end{bmatrix} \geq 0. \quad (9)$$

In [1], it is shown that for $p = p_n/p_d > 1$, we have $k(p_n, p_d) \leq \log_2 p_n + \alpha(p_n)$, where $k(p_n, p_d)$ is the number of LMI constraints that are generated to represent $x^3 \leq y$, and $\alpha(p_n)$ is a term that grows slowly compared to $\log_2$ term. Therefore, it is beneficial to use as few basis functions as possible to efficiently compute the solutions of the DC problems because we need extra variables and constraints to represent the functions y^p . Therefore, Chebyshev polynomials become a rather inefficient choice, as they tend to be dense polynomials with high degrees, which is required for accurate approximation. Recall that, Figure 1 shows the Chebyshev approximation diverges, when the error tolerance is set to be small.

Example 2: Consider the MDP in Figure 4. The DC problem that computes the maximum probability of satisfying the specification, given a posynomial with K basis functions,

$$\begin{aligned} \text{maximize}_{\sigma(s,a), \sigma(s,b)} \quad & \sum_{k=1}^K \left(c_k \cdot \left((\sigma(s,a) + \sigma(s,b))^{a_k} \cdot (u_+(0.2)) \right. \right. \\ & + \left(\sigma(s,a) + 0.4 \cdot \sigma(s,b) \right)^{a_k} \cdot (u_+(0.5) - u_+(0.2)) + \\ & \left. \left. (0.4 \cdot \sigma(s,b))^{a_k} \cdot (u_+(0.9) - u_+(0.5)) \right) \right) \end{aligned}$$

subject to $\sigma(s,a) + \sigma(s,b) = 1, \sigma(s,a) \geq 0, \sigma(s,b) \geq 0$.

We note that the objective in the above problem is a sum of DC functions and the functions in the constraints are affine functions, the above problem is a DC optimization problem and a locally optimal solution of the problem can be computed using Algorithm 1.

So far, we considered formal quantitative verification of the systems, and these problems do not include cost or reward function. If we want to include cost or reward functions in a MDP to minimize the expected cost or maximize the expected reward with CPT-based measures, then objective of the optimization problem in (7) will be replaced by the following:

$$\begin{aligned} \text{maximize}_{\sigma(s,a), a \in Act} \quad & \sum_{s'_q \in S(s)} \left(\Phi'_q \cdot \left(u_+(\gamma_s + v_{t+1}(s'_q)) \right. \right. \\ & \left. \left. - u_+(\gamma_s + p_{t+1}(v'_{q-1})) \right) \right) \\ \gamma_s &= \sum_{k=1}^K c_k \cdot \left(\sum_{\alpha \in Act(s)} \sigma(s, \alpha)^{a_k} \cdot R_t(s, \alpha) \right), \end{aligned} \quad (10)$$

where $v_t(s'_q)$ denotes the value of a state with index q at time t .

Note that, the term $u_+(\gamma_s + v_{t+1}(s'_q))$ is a composition two convex or concave functions, which is not convex or concave in general, also that term is multiplied with a DC function Φ_q . To the best of our knowledge, no general method exists to solve problems with this type of objective. But for two special cases, we can efficiently compute locally optimal solutions using CCP. If the cost or reward function is a function of state instead of state and action, then we can modify the objective function in (8) as:

$$\begin{aligned} \text{maximize}_{\sigma(s,a), a \in Act} \quad & \sum_{s'_q \in S(s)} \left(\Phi'_q \cdot \left(u_+(C(s) + v_{t+1}(s'_q)) \right. \right. \\ & \left. \left. - u_+(C(s) + v_{t+1}(s'_{q-1})) \right) \right). \end{aligned} \quad (11)$$

As the cost is a constant, the objective in (11) is a sum of DC functions, therefore we can compute the locally optimal solution for the case when the cost or reward function is function of a state.

The second special case we consider is when the utility functions are $u_-(x) = u_+(x) = x$. Then, adding γ_s to the objective term in (8) will result in a formulation that computes the optimal policy for this special case.

V. NUMERICAL RESULTS

We demonstrate the proposed approach on three domains: (1) A robot in a gridworld, (2) a consensus protocol, and (3) a ride sharing example. The simulations were performed on a computer with an Intel Core i5-7200u 2.50 GHz processor and 8 GB of RAM with MOSEK [2] as solver and using the CVX [15] interface.

A. Grid world

Consider a grid world, where states are defined as grid points on a map. An agent starts in an initial state and the objective of the agent is to reach to a given state with minimal cost. The agent can move in four directions by selecting actions (north, south, east, and west). The probability of arriving at the intended cell is δ , and with probability $1 - \delta$, the agent moves to a random neighboring state. The cost in each move until reaching the destination is 1 for each state. The grid world has a number of static obstacles and the agent has to avoid these obstacles as hitting an obstacle has a high cost of $M = 50$. Therefore, the objective is to compute a safe (i.e., not hitting to obstacles) path that is cost efficient. In our experiments, we choose a 50×50 grid-world, and the gridworld MDP has 2,500 states with horizon $T = 100$, $\delta = 0.2$ and 300 states consisting of obstacles that the agent tries to evade. We use the weighting function $\exp(-0.5(-\ln(p))^{0.9})$ and the utility function $x^{0.88}$.

After obtaining the policies using Algorithm 1, we evaluate the policies on 500 simulation runs. The risk-neutral policy finds a shorter route (with average cost equal to 38.137 on successful runs), yet it crashes into obstacles in 41 runs. In contrast, the risk-averse policy chooses longer routes (with average cost equal to 57.638 on successful runs), but it crashes into obstacles only in 6 runs.

B. Consensus Protocol

This case study deals with modeling and verifying the shared coin protocol of the randomized consensus algorithm of Aspnes and Herlihy [4]. The shared coin protocol returns a preference 1 or 2, with a certain probability, whenever requested by a process at some point in the execution of the consensus algorithm. It implements a collective random walk parameterized by the number of processes N and the constant $K > 1$ (independent of N).

The first property that we want to compute is the minimal probability of finishing the process and all processes being 1, which can be expressed as maximizing the probability states, where the execution is finished, and all coins will have the value 1, after the process. For each benchmark instance, Figure 5 gives the number of states (#states), the computation time for CPT-based measures, the minimum probability of satisfying the specification with CPT-based measure (CPT $\mathbb{P}$) and the actual minimum probability of satisfying the specification $\mathbb{P}$ in the model. We use the same weighting and utility function as in the previous example.

We considered the verification of another property, where we want to compute the maximum probability of finishing

Parameters	#states	Time (s)	CPT $\mathbb{P}$	$\mathbb{P}$
$K = 2$	272	34.49	0.615	0.383
$K = 16$	2064	384.93	0.722	0.484
$K = 64$	8208	1961.34	0.673	0.498

Fig. 5: Results for consensus benchmark with the property of all coins having the same value.

the process and all coins not having the same value. Figure 6 shows the results for each instance.

Parameters	#states	Time (s)	CPT $\mathbb{P}$	$\mathbb{P}$
$K = 2$	272	41.68	0.315	0.108
$K = 16$	2064	472.19	0.212	0.016
$K = 64$	8208	2953.75	0.163	0.002

Fig. 6: Results for consensus benchmark the property of all coins not having the same value.

Both examples in consensus protocol shows, the CPT-based measure tends to inflate the probability of satisfying the properties. The weighting function overestimates the small probabilities of the transition probabilities in MDPs and the utility function that we choose inflates the reachability probabilities.

C. Ride Sharing

We consider a ride sharing example, inspired by [31]. This case study concerns modeling the behavior of a passenger in a sequential decision-making scenario. Many ride-sharing companies set prices on their rides based on both supply of drivers and demand of passengers. Therefore, the price of a ride may fluctuate. The passengers account for the price fluctuation, which influences their behavior.

We model the ride-sharing MDP with $S = \{0, 1, 2, 3, 4\}$, where states 0, ..., 3 denote the cases where the passenger did not take a ride and state 4 represents the case when the passenger takes a ride. The price multipliers for states 0, 1, 2 and 3 are 1.0, 1.4, 1.8 and 2.2 respectively. $Act = \{0, 1\}$ where action 0 is waiting, and action 1 is taking a ride. We consider a horizon length $T = 5$, and the transition matrix for action 0 is

$$\mathcal{P} = \begin{bmatrix} 0.876 & 0.099 & 0.017 & 0.008 \\ 0.347 & 0.412 & 0.167 & 0.074 \\ 0.106 & 0.353 & 0.259 & 0.282 \\ 0.086 & 0.219 & 0.143 & 0.552 \end{bmatrix}.$$

If action 1 is taken, the passenger transitions to state 4 with probability 1, which implies that a ride has been taken. We define the reward function as

$$R(s_t, a_t) = \begin{cases} \hat{R} & a_t = 0, \\ S_t - x_t(p_{base} + p_{mile}D + p_{min}T) & a_t = 1, \end{cases}$$

where $\hat{R}$ is a constant, D is the distance in miles, T is time in minutes, S_t is a constant that decreases linearly in time, x_t is the price multiplier, and p_{base} , p_{mile} , and p_{min} are the base, per mile, and per min prices, respectively. We choose

the prices based on Uber’s Washington, DC operation¹, and we use the same weighting function as in the previous examples and utility function $u_+(x) = x$. Table 1 shows the probabilities of taking a ride at a price multiplier and time. We note that our passenger model is relatively risk-

TABLE I: Probabilities of taking a ride with respect to the price multiplier and time.

Price multiplier	1	1.4	1.8	2.2
Time				
1	0.88	0.25	0.17	0.13
2	0.94	0.89	0.56	0.45
3	0.97	0.83	0.82	0.78
4	0.99	0.95	0.95	0.86
5	0.99	0.99	0.98	0.98

averse, i.e., the probability of taking a ride is very high when the price multiplier is 1, and the probability decreases with increasing price multipliers. The passengers tend to take a ride with increasing time to avoid taking any further risks in case of an increase in price multiplier in the future.

VI. CONCLUSIONS

We proposed a computational method for verification of temporal logic specifications in Markov decision processes (MDPs) with measures from cumulative prospect theory (CPT). CPT-based measures are empirically known to faithfully capture the asymmetry in the risk-averseness and risk-taking behavior of humans in decision-making. Computation of optimal policies is impractical with CPT-based measures due to the nonlinear weighting and utility functions. The proposed method approximates the nonlinear weighting function with a difference of convex (DC) function, then computes a locally optimal policy by solving a DC problem. On the other hand, computing a policy with a CPT-based measure takes more time than computing a policy with expected-value measure, as we need to represent the DC functions as a series of linear matrix inequalities. We demonstrate the practical applicability of our approach on several scenarios. For future work, we are interested in establishing error bounds between the globally optimal CPT-value and the CPT-value that is obtained from our method in MDPs.

REFERENCES

- [1] F. Alizadeh and D. Goldfarb. Second-order cone programming. *Mathematical Programming*, 95(1):3–51, 2003.
- [2] E. D. Andersen and K. D. Andersen. The MOSEK optimization software. *EKA Consulting ApS, Denmark*, 2012.
- [3] I. P. Androulakis, C. D. Maranas, and C. A. Floudas. α bb: A global optimization method for general constrained nonconvex problems. *Journal of Global Optimization*, 7(4):337–363, 1995.
- [4] J. Aspnes and M. Herlihy. Fast randomized consensus using shared memory. *Journal of Algorithms*, 11(3):441–461, 1990.
- [5] C. Baier and J.-P. Katoen. *Principles of Model Checking*. MIT Press, 2008.
- [6] N. C. Barberis. Thirty years of prospect theory in economics: A review and assessment. *The Journal of Economic Perspectives*, 27(1):173–195, 2013.
- [7] D. P. Bertsekas and J. N. Tsitsiklis. An analysis of stochastic shortest path problems. *Mathematics of Operations Research*, 16(3):580–595, 1991.

- [8] T. Bielecki, D. Hernández-Hernández, and S. R. Pliska. Risk-sensitive control of finite state Markov chains in discrete time, with applications to portfolio management. *Mathematical Methods of Operations Research*, 50(2):167–188, 1999.
- [9] C. D. Boor. *A Practical Guide to Splines*, volume 27. Springer-Verlag New York, 1978.
- [10] V. S. Borkar. Q-learning for risk-sensitive control. *Mathematics of Operations Research*, 27(2):294–311, 2002.
- [11] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.
- [12] Y. Chow and M. Ghavamzadeh. Algorithms for CVaR optimization in MDPs. In *Advances in Neural Information Processing Systems*, pages 3509–3517, 2014.
- [13] Y. Chow, A. Tamar, S. Mannor, and M. Pavone. Risk-sensitive and robust decision-making: A CVaR optimization approach. In *Advances in Neural Information Processing Systems*, pages 1522–1530, 2015.
- [14] J. A. Filar, D. Krass, and K. W. Ross. Percentile performance criteria for limiting average Markov decision processes. *IEEE Transactions on Automatic Control*, 40(1):2–10, 1995.
- [15] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. <http://cvxr.com/cvx>, Mar. 2014.
- [16] D. Hernández-Hernández and S. I. Marcus. Risk-sensitive control of Markov processes in countable state space. *Systems & Control Letters*, 29(3):147–155, 1996.
- [17] R. A. Howard and J. E. Matheson. Risk-sensitive Markov decision processes. *Management Science*, 18(7):356–369, 1972.
- [18] J.-P. Katoen. The probabilistic model checking landscape. *IEEE Symposium on Logic In Computer Science*, 2016.
- [19] M. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: Verification of probabilistic real-time systems. *Computer Aided Verification*, 6806:585–591, 2011.
- [20] E. L. Lawler and D. E. Wood. Branch-and-bound methods: A survey. *Operations Research*, 14(4):699–719, 1966.
- [21] K. Lin. *Stochastic Systems with Cumulative Prospect Theory*. PhD thesis, University of Maryland, College Park, 2013.
- [22] K. Lin and S. I. Marcus. Dynamic programming with non-convex risk-sensitive measures. In *American Control Conference (ACC)*, pages 6778–6783. IEEE, 2013.
- [23] T. Lipp and S. Boyd. Variations and extension of the convex-concave procedure. *Optimization and Engineering*, 17(2):263–287, 2016.
- [24] L. L. Lopes and G. C. Oden. The role of aspiration level in risky choice: A comparison of cumulative prospect theory and SP/A theory. *Journal of Mathematical Psychology*, 43(2):286–313, 1999.
- [25] J. C. Mason and D. C. Handscomb. *Chebyshev Polynomials*. CRC Press, 2002.
- [26] J. Nocedal and S. J. Wright. *Sequential Quadratic Programming*. Springer, 2006.
- [27] M. Ono, M. Pavone, Y. Kuwata, and J. Balaram. Chance-constrained dynamic programming with application to risk-aware robotic space exploration. *Autonomous Robots*, 39(4):555–571, 2015.
- [28] L. Prashanth, C. Jie, M. Fu, S. Marcus, and C. Szepesvári. Cumulative prospect theory meets reinforcement learning: Prediction and control. In *International Conference on Machine Learning*, pages 1406–1415, 2016.
- [29] D. Prelec. The probability weighting function. *Econometrica*, pages 497–527, 1998.
- [30] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2014.
- [31] L. J. Ratliff and E. Mazumdar. Risk-sensitive inverse reinforcement learning via gradient methods. *arXiv preprint arXiv:1703.09842*, 2017.
- [32] A. Ruszczyński. Risk-averse dynamic programming for Markov decision processes. *Mathematical Programming*, 125(2):235–261, 2010.
- [33] X. Shen, S. Diamond, Y. Gu, and S. Boyd. Disciplined convex-concave programming. In *Conference on Decision and Control (CDC)*, pages 1009–1014. IEEE, 2016.
- [34] A. Tamar, Y. Glassner, and S. Mannor. Optimizing the CVaR via sampling. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.
- [35] A. Tversky and D. Kahneman. Advances in prospect theory: Cumulative representation of uncertainty. *Journal of Risk and Uncertainty*, 5(4):297–323, 1992.

¹<http://uberestimate.com/prices/Washington-DC/>