ROBUST CONTROL OF MARKOV DECISION PROCESSES: LITERATURE REVIEW AND APPLICATIONS

AYAN BISWAS, ZHE CAO, HAOTIAN GU, JIAYU YE, AND SHIZHAN ZHU
UC BERKELEY

Abstract. Markov Decision problems are encountered frequently in real-life applications. The problems are often prone to uncertainties in environment dynamics. In many practical problems, ignorance of potential risk limits the performance of the optimal policy. This is a well explored field with lots of rich literature. We reviewed some relevant papers and summarized the different formulations of the Markov Decision Process, specifically focusing on uncertainty pertaining to two settings: uncertainty in reward/cost and uncertainty in transition dynamics. We conducted two numerical experiments to verify the efficacy of uncertainty set modelling. The first experiment is the classic aircraft path routing problem when the storm map in the environment follows an uncertain Markov Process. The other is the machine replacement problem, with uncertain replacement cost and uncertain transition matrix. Our experimental results showed the general improvement of the optimized performance when uncertainty sets are modelled with robust or chance-constraint formulations.

1. Overview

The report is organized as follows. In Sec. 2, we introduce the theory of Markov Decision processes with finite and infinite time horizons. Sec. 3 presents the general problem formulation of Markov Decision Processes with uncertain costs, while Sec. 4 elaborates the uncertain transition parameters case. In Sec. 5, we dive into the first application of Aircraft Routing Problem, while in Sec. 6 we present the experimental results for the Machine Replacement problem. Finally we conclude the paper in Sec. 7.

2. Markov Decision Processes

2.1. **Problem Formulation.** A Markov Decision Process $M = (\mathcal{S}, \mathcal{A}, T, P, C, \gamma)$ consists of the following components:

- a finite state space $\mathcal{S}$
- a finite action space $\mathcal{A}$
- a finite or infinite time horizon T
- a collection of transition matrices $\{P_t^a\}_{a\in\mathcal{A},t=0\ldots T-1}$ where $P_t^a(i,j)$ is the probability of transiting from state i to j when playing action a at time t
- a collection of cost functions $\{C_t\}_{t=0\ldots T}$ where $C_t(i,a)$ corresponds to a cost the player needs to pay when playing action a at state i at time t, and $C_T(i)$ corresponds to a terminal cost
- a discount factor γ when T is infinite

We use q to denote a distribution on A which is satisfied by the initial state i_0.

A policy $\pi = (\pi_0 \dots \pi_{T-1}) \in \Pi$ can be viewed as a function that specifies the action that the player will choose when in state s at time t. The goal is to find π to minimize the cost over T. There are several types of policies studied in this report.

- A policy is called deterministic if each π_t is a mapping from $\mathcal{S}$ to $\mathcal{A}$. We denote the set of deterministic policies by Π_d.
- A policy is called stochastic if each π_t is a mapping from $\mathcal{S}$ to a probability distribution on $\mathcal{A}$. We denote the set of stochastic policies by Π_s.
- A policy is called stationary if π_t is invariant over time t. We use Π^s to denote the set of stationary policies.

When T is finite, given $M = (\mathcal{S}, \mathcal{A}, T, P, C)$, the nominal MDP is to find

$$\phi_T(\Pi, P) := \min_{\pi \in \Pi} C_T(\pi, P) := \min_{\pi \in \Pi} \mathbb{E}_{\pi,P} \left[\sum_{t=0}^{T-1} C_t(i_t, \pi_t(i_t)) + C_T(i_T) \right]. \tag{2.1}$$

When T is infinite, we often consider the stationary cost functions and transition matrices, which means $C_t = C, P_t^a = P^a$ for any t. In this case, we restrict ourselves to consider the stationary policies. So the nominal problem is

$$\phi_\infty(\Pi^s, P) := \min_{\pi \in \Pi^s} C_\infty(\pi, P) := \min_{\pi \in \Pi^s} \mathbb{E}_{\pi,P} \left[\sum_{t=0}^{\infty} \gamma^t C(i_t, \pi(i_t)) \right]. \tag{2.2}$$

2.2. **Markov Decision Processes with Finite Time Horizon.**

2.2.1. *Dynamic Programming: Policy Iteration and Value Iteration.* To get an optimal policy in Π_d for the problem 2.1, we can apply the following dynamic programming techniques. Let $v_t(i)$ be the optimal value function at time t and state i.

$$v_t(i) = \min_{a \in \mathcal{A}} \left[C_t(i, a) + \sum_{j \in \mathcal{S}} P_t^a(i, j) v_{t+1}(j) \right] \tag{2.3}$$

$$\pi_t(i) = \arg\min_{a \in \mathcal{A}} \left[C_t(i, a) + \sum_{j \in \mathcal{S}} P_t^a(i, j) v_{t+1}(j) \right] \tag{2.4}$$

2.2.2. *Linear Programming Formulation.* 2.1 can also be formulated as a LP and the detailed discussion can be found in [5].

$$\begin{aligned} &\max_{v_0 \dots v_{N-1}} \sum_{i \in \mathcal{S}} q(i) v_0(i) \\ \text{s.t. } v_t(i) &\le C_t(i, a) + \sum_{j \in \mathcal{S}} P_t(j|i, a) v_{t+1}(j) \quad \forall a \in \mathcal{A}, i \in \mathcal{S}, t = 0 \dots T-1 \end{aligned} \tag{2.5}$$

2.3. **Markov Decision Processes with Infinite Time Horizon.**

2.3.1. *Bellman Recursion: Policy Iteration and Value Iteration.* Let $v(i)$ be the optimal value function at state i. Then the optimality condition is

$$v(i) = \min_{a\in\mathcal{A}} \left[C(i,a) + \gamma \sum_{j\in\mathcal{S}} P^a(i,j)v(j) \right]. \tag{2.6}$$

Define the Bellman operator T on $\mathbb{R}^{|\mathcal{S}|}$, such that for any $u \in \mathbb{R}^{|\mathcal{S}|}$,

$$Tu(i) = \min_{a\in\mathcal{A}} \left[C(i,a) + \gamma \sum_{j\in\mathcal{S}} P^a(i,j)u(j) \right]. \tag{2.7}$$

The optimal value function v is the unique fixed point of T and it can be obtained by fixed point iterations. Furthermore, an optimal policy can be obtained by

$$\pi(i) = \arg\min_{a\in\mathcal{A}} \left[C(i,a) + \gamma \sum_{j\in\mathcal{S}} P^a(i,j)v(j) \right]. \tag{2.8}$$

2.3.2. *Linear Programming Formulation.* 2.2 can also be formulated as a LP and the detailed discussion can be found in [5].

$$\begin{aligned} &\max_{v} \sum_{i\in\mathcal{S}} q(i)v(i) \\ &\text{s.t. } v(i) \le C(i,a) + \gamma \sum_{j\in\mathcal{S}} P(j|i,a)v(j) \quad \forall a \in \mathcal{A}, i \in \mathcal{S} \end{aligned} \tag{2.9}$$

3. Markov Decision Processes Under Uncertain Costs

In the previous discussion, we see how the nominal MDP problem is solved by dynamic programming or linear programming. But in real-world applications, it is natural to encounter situations where costs or transition probabilities are uncertain. In this section, we will focus on solving MDP problems under uncertain costs. To simplify the discussion, we will only consider the infinite horizon MDP, but the methodologies we introduce can be easily extended to the finite horizon case. We mainly refer to [2] and [7] for results in this section.

3.1. **Problem Formulation.**

There are different ways to model the uncertainty on cost functions. For example, we can assume now the cost functions are random variables, $C \sim \mu$, where μ can either be a known distribution or an unknown distribution belonging to some class U. For the first case, as also mentioned in the lecture, one usual approach is to add a chance constraint to the nominal problem 2.2:

$$\begin{aligned} &\min_{\pi\in\Pi^s, y} y \\ &\text{s.t. } \mathbb{P}_C \left(\mathbb{E}_{\pi,P} \left[\sum_{t=0}^{\infty} \gamma^t C(i_t, \pi(i_t)) \right] \le y \right) \ge 1-\epsilon. \end{aligned} \tag{3.1}$$

For the second case, we can consider the distributionally robust policy

$$\min_{\pi\in\Pi^s} \max_{C\sim\mu\in U} \mathbb{E}_{\pi,P} \left[\sum_{t=0}^{\infty} \gamma^t C(i_t, \pi(i_t)) \right] \tag{3.2}$$

And this formulation is studied in [7]. In the remaining part of this section, we will focus on the first case.

3.2. **Chance Constraint: Theory and Derivation.** At a first glance, solving 3.1 seems to be complicated. However, after dualizing 2.9, we arrive at the following LP:

$$\begin{aligned} &\min_{\rho} \sum_{i\in\mathcal{S},a\in\mathcal{A}} \rho(i,a)C(i,a) \\ \text{s.t. } & q(i) - \sum_{a\in\mathcal{A}} \rho(i,a) + \gamma \sum_{j\in\mathcal{S},a\in\mathcal{A}} P(i|j,a)\rho(j,a) = 0, \\ &\rho \geq 0. \end{aligned} \tag{3.3}$$

Note that in this form, the only uncertainty is in the objective. Hence, the chance programming 3.1 is equivalent to

$$\begin{aligned} &\min_{\rho,y} y \\ \text{s.t. } & \mathbb{P}_C \left(\sum_{i\in\mathcal{S},a\in\mathcal{A}} \rho(i,a)C(i,a) \leq y \right) \geq 1-\epsilon \\ & q(i) - \sum_{a\in\mathcal{A}} \rho(i,a) + \gamma \sum_{j\in\mathcal{S},a\in\mathcal{A}} P(i|j,a)\rho(j,a) = 0, \\ &\rho \geq 0. \end{aligned} \tag{3.4}$$

In general, there are a rich theory on how to deal with the chance constraint in 3.4. For example, in the lecture, we have studied how to build the Bernstein approximation to turn the problem into a tractable form. Meanwhile, for some special classes of distributions, we can further simplify the problem. One case is presented in the next subsection.

The other remark is that, after solving 3.4 and get ρ^*, we still want to recover an optimal condition. In [5], it has been proved that there is a 1-1 correspondence between the feasible solution ρ and the policy space Π_s^s. More explicitly, we can define the optimal policy π^* by

$$\pi^*(i,a) = \begin{cases} \frac{1}{|\mathcal{A}|} & \text{if } \sum_{a\in\mathcal{A}} \rho^*(i,a) = 0 \\ \frac{\rho^*(i,a)}{\sum_{a\in\mathcal{A}} \rho^*(i,a)} & \text{otherwise} \end{cases} \tag{3.5}$$

3.3. **Case Study: Cost Uncertainty with Gaussian Distribution.** Now assume C satisfies the multivariate Gaussian distribution $\mathcal{N}(\mu_C, \Theta_C)$ Under this setting, it is well-known that the constraint

$$\mathbb{P}_C \left(\sum_{i\in\mathcal{S},a\in\mathcal{A}} \rho(i,a)C(i,a) \leq y \right) \geq 1-\epsilon \tag{3.6}$$

is equivalent to

$$\sum_{i\in\mathcal{S},a\in\mathcal{A}} \rho(i,a)\mu_C(i,a) + \Phi^{-1}(1-\epsilon)\sqrt{\rho^T\Theta_C\rho} \leq y \tag{3.7}$$

Hence, if C is Gaussian, then this problem is equivalent to an SOCP.

One simple generalization of 3.6 is that, instead of assuming $C \sim \mathcal{N}(\mu_C, \theta_C)$, we can consider the moment constraints

$$\begin{gathered} \mathbb{P}_C \left(\sum_{i \in \mathcal{S}, a \in \mathcal{A}} \rho(i,a) C(i,a) \leq y \right) \geq 1 - \epsilon, \\ \forall C \text{ s.t. } \mathbb{E}[C] = \mu_C,\ \mathbb{E}\left[(C - \mu_C)(C - \mu_C)^T \right] = \Theta_C \end{gathered} \tag{3.8}$$

It is also a well-known fact that 3.8 is equivalent to

$$\sum_{i \in \mathcal{S}, a \in \mathcal{A}} \rho(i,a) \mu_C(i,a) +] \sqrt{\frac{1-\epsilon}{\epsilon}} \sqrt{\rho^T \Theta_C \rho} \leq y. \tag{3.9}$$

More detailed discussions can be found in [1].

4. Markov Decision Processes Under Uncertain Transition Parameters

In this section, we will focus on dealing with uncertain transition parameters. As before, we will still present results only for the infinite horizon MDP, and these results can be extended to the finite horizon case easily. We mainly refer to [4] and [6].

4.1. **Problem Formulation.**

There are several different approaches to model the uncertainty on transition parameters. Similar to what we have discussed in the previous section, one possible formulation is to assume transition parameters are random variables and then to introduce chance constraints on transition matrices. This problem is studied in [2]. But it turns out that the chance programming on P is in general very difficult to deal with, even when we only consider the Dirichlet random variables. Some computationally tractable approximation approaches are proposed to handle this problem. See [2] for more details.

Another possible formulation is that, for each action a, the corresponding transition matrix P^a is only known to lie in some set of confidence $\mathcal{P}^a$. $P = \{P^a\}_{a \in \mathcal{A}} \in \bigotimes_{a \in \mathcal{A}} \mathcal{P}^a := \mathcal{P}$. The robust counterpart of 2.2 is stated as

$$C^* = \min_{\pi \in \Pi^s} \max_{P \in \mathcal{P}} \mathbb{E}_{\pi, P} \left[\sum_{t=0}^{\infty} \gamma^t C(i_t, \pi(i_t)) \right]. \tag{4.1}$$

If $\mathcal{P}$ is a confidence region that contains the true transition parameter P with probability $1 - \beta$, then we want to find a policy π that guarantees an expected total cost of at most C^* at a confidence level $1 - \beta$.

In [4], this problem has been studied, for both the finite horizon and the infinite horizon, under the so-called 'rectangular uncertainty property'. More specifically, it means that for every $a \in \mathcal{A}$, $\mathcal{P}^a$ has the form $\mathcal{P}^a = \bigotimes_{i \in \mathcal{S}} \mathcal{P}_i^a$, where $\mathcal{P}_i^a$ is some given subset of the probability simplex in $\mathbb{R}^{|\mathcal{S}|}$ that describe the uncertainty on the state distribution given action a at state i.

In remaining part of this section, we will first follow the framework of [4] and see how 4.1 is solved by the robust version of Bellman recursion. We also notice that in [6], another class of uncertainty set which may not satisfy the rectangular uncertainty assumption is studied. We will also briefly mention some interesting results from [6]. In fact, for a general non-rectangular uncertainty set, it has been proved in [6] that the robust MDP problem cannot be approximated to any constant factor in polynomial time.

4.2. **Robust Bellman Recursion: Theory and Derivation.** The main result in [4] is the following.

Theorem 4.1. *Under the rectangular uncertainty property for the infinite horizon robust MDP problem 4.1, the strong duality holds:*

$$\min_{\pi\in\Pi^s}\max_{P\in\mathcal{P}}\mathbb{E}_{\pi,P}\left[\sum_{t=0}^{\infty}\gamma^t C(i_t,\pi(i_t))\right]=\max_{P\in\mathcal{P}}\min_{\pi\in\Pi^s}\mathbb{E}_{\pi,P}\left[\sum_{t=0}^{\infty}\gamma^t C(i_t,\pi(i_t))\right] \tag{4.2}$$

Moreover, let $v(i)$ be the optimal value function at state i. Then the optimality condition is

$$v(i)=\min_{a\in\mathcal{A}}\left[C(i,a)+\gamma\sigma_{\mathcal{P}_i^a}(v)\right]. \tag{4.3}$$

Here $\sigma_{\mathcal{P}_i^a}(v):=\sup\{p^Tv:p\in\mathcal{P}_i^a\}$. Define the Bellman operator T on $\mathbb{R}^{|\mathcal{S}|}$, such that for any $u\in\mathbb{R}^{|\mathcal{S}|}$,

$$Tu(i)=\min_{a\in\mathcal{A}}\left[C(i,a)+\gamma\sigma_{\mathcal{P}_i^a}(u)\right]. \tag{4.4}$$

The optimal value function v is the unique fixed point of T and it can be obtained by fixed point iterations. Furthermore, an optimal policy can be obtained by

$$\pi(i)=\mathrm{argmin}_{a\in\mathcal{A}}\left[C(i,a)+\gamma\sigma_{\mathcal{P}_i^a}(v)\right]. \tag{4.5}$$

Here we omit the detailed proof of this theorem. But the basic idea is to take advantage of the LP formulation 2.9.

Note that in order to conduct the fixed point iteration, we need to solve the inner problem

$$\sup\{p^Tv:p\in\mathcal{P}_i^a\} \tag{4.6}$$

at each step. It can be imagined that whether or not this inner problem is easy to handle depends on how $\mathcal{P}_i^a$ is chosen. One of the simplest models is that P_i^a lies inside some given intervals of confidence component-wise. In this case, the inner problem is just a LP. Here we introduce another 2 models for constructing $\mathcal{P}_i^a$, with their statistical interpretations and numerical treatments.

4.3. **Case Study: Likelihood Model.**

4.3.1. *Model Description.* For any action a, assume we have a matrix of empirical frequencies of transition with action a in the experiment, which is denoted by F^a. Then F^a is the solution of the following maximum-likelihood problem:

$$\begin{aligned}&\max_{P^a}\sum_{i,j\in\mathcal{S}}F^a(i,j)\log P^a(i,j)\\ &\text{s.t. } P^a\mathbf{1}=\mathbf{1},\ P^a(i,j)\geq 0,\ \forall i,j\in\mathcal{S}.\end{aligned} \tag{4.7}$$

Therefore, we can define a likelihood uncertainty set for P^a by

$$\mathcal{P}^a:=\left\{P^a\in\mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}:P^a\mathbf{1}=\mathbf{1},\ P^a(i,j)\geq 0,\ \sum_{i,j\in\mathcal{S}}F^a(i,j)\log P^a(i,j)\geq\beta^a\right\} \tag{4.8}$$

Here the parameter β^a is chosen to describe the uncertainty level of our empirical estimator F^a. In fact, it is a well-known statistical result that for asymptotically large samples, there is a one-to-one correspondence between the level of confidence and the lower bound β^a. See, for example, [3] for details.

Note that in this case 4.8, $\mathcal{P}^a$ doesn't satisfy the rectangular assumption. So we define $\mathcal{P}_i^a$ by projecting $\mathcal{P}^a$ onto the i^{th} row of the matrix. This will yield

$$\mathcal{P}_i^a := \left\{ p \in \mathbb{R}^{|\mathcal{S}|} : p^T \mathbf{1} = 1,\ p \geq \mathbf{0},\ \sum_{j \in \mathcal{S}} F^a(i,j) \log p(j) \geq \beta^a - \sum_{k \neq i} \sum_{j \in \mathcal{S}} F^a(k,j) \log F^a(k,j) \right\} \tag{4.9}$$

4.3.2. *Finding Support Functions.* Note that, in general, the inner problem 4.6 is of the form

$$\begin{aligned} &\max_p\ p^T v \\ &\text{s.t. } p^T \mathbf{1} = 1,\ p \geq \mathbf{0},\ \sum_j f(j) \log p(j) \geq \beta. \end{aligned} \tag{4.10}$$

This problem can be solved by the following procedure.

First, consider the dual problem of 4.10 is

$$\begin{aligned} &\min_{\lambda,\mu,\zeta}\ \mu - (1+\beta)\lambda + \lambda \sum_j f(j) \log \frac{\lambda f(j)}{\mu - v(j) - \zeta(j)} \\ &\text{s.t. } \lambda \geq 0,\ \zeta \geq \mathbf{0},\ \zeta + v \leq \mu \mathbf{1}. \end{aligned} \tag{4.11}$$

Since the problem is convex and Slater's condition is satisfied, then there is no duality gap. By a monotonicity argument, we can show that the optimal choice for ζ is 0. Moreover, for a fixed μ, the optimal choice for λ can be explicitly determined. This reduces the original problem to a one dimensional convex optimization problem, which can be solved efficiently by simple numerical optimization methods, such as the bisection algorithm.

4.4. **Case Study: Entropy Model.**

4.4.1. *Model Description.* Another way to construct the uncertainty set is to use the Kullback-Leibler divergence.

$$\mathcal{P}_i^a := \left\{ p \in \mathbb{R}^{|\mathcal{S}|} : p^T \mathbf{1} = 1,\ p \geq \mathbf{0},\ \sum_{j \in \mathcal{S}} p(j) \log \frac{p(j)}{q_i^a(j)} \leq \beta_i^a \right\} \tag{4.12}$$

Here q_i^a is some known distribution on $\mathcal{S}$

4.4.2. *Finding Support Functions.* Similar as the likelihood model, the inner problem can be converted to a one-dimensional optimization problem by the duality and convexity argument. More detailed derivation can be found in [4].

4.5. **S-rectangular Uncertainty Set.** Instead of the rectangular uncertainty assumption, in [6], a class of so called 's-rectangular' uncertainty sets is studied. More specifically, assume there exists a subset $U \in \mathbb{R}^q$, and for any $a \in \mathcal{A}$ and $i \in \mathcal{S}$, there exist some $k_{ia} \in \mathbb{R}^{|\mathcal{S}|}$ and $K_{ia} \in \mathbb{R}^{|\mathcal{S}| \times q}$. For any state i, define the set

$$\mathcal{P}_i = \{\{P_u^a(i,\cdot)\}_{a \in \mathcal{A}} : u \in U\},\ P_u^a(i,\cdot) = k_{ia} + K_{ia} u \tag{4.13}$$

and the uncertainty set is defined to be

$$\mathcal{P} := \bigotimes_{i \in \mathcal{S}} \mathcal{P}_i \tag{4.14}$$

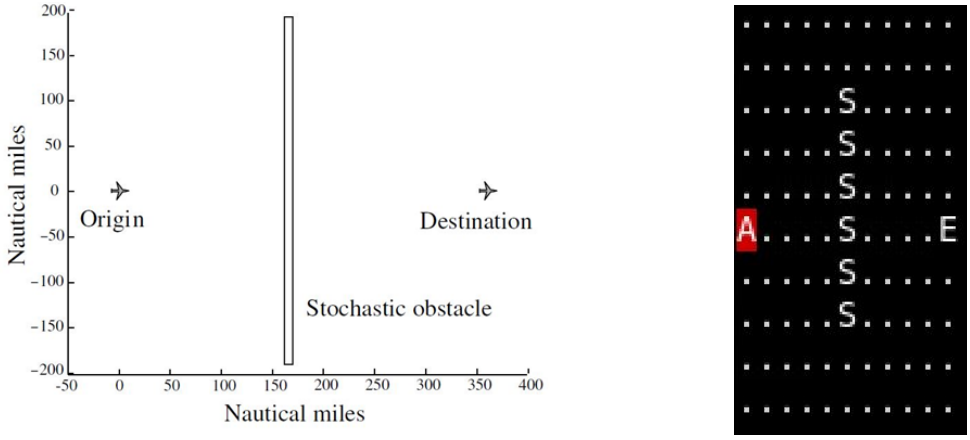


FIGURE 1. Graphical representation of air-space grid [4], and our implementation

Note that under the previous rectangular assumption, all transition probabilities $P^a(s,\cdot)$ can vary freely in the set $\mathcal{P}_i^a$. But under s-rectangular assumption, different actions in the same state may be dependent. This dependence may arise when the actions relate to varying degrees of intensity with which a task is executed. The rectangular assumption implies the s-rectangular assumption.

Moreover, it is proved in [6] that if U is a finite intersection of closed halfspaces and ellipsoids, then the robust MDP 4.1 can be solved by a similar fixed point iteration scheme as we have discussed in this section, and for each iteration, we need to solve an inner problem by SOCP and SDP. Overall, this problem is tractable.

5. Application: Aircraft Routing Problem

5.1. **Problem Formulation.** We implement and verify the results pertaining to the robust aircraft routing problem described in [4]. In this problem, the air space is visualized as a rectangular grid, where every node represents a state in the state vector. The airplane has to travel from the origin to the destination by incurring the minimum delay, which is the cost function. There is a storm (obstacle) map that evolves following a Markov model. The transition matrix of the Markov process is constant in the nominal scenario, but follows a random distribution in the perturbed version. If there are k obstacles, then the Markov chain has a $2^k \times 2^k$ transition matrix.

The air-space grid, and its corresponding implementation, are shown in Fig 1. The current position of the plane is represented by A (marked in red), which flies to the end state represented by E. The presence of a storm is denoted by S.

For simplicity, we assume that the plane can only travel horizontally and vertically, i.e. there are 4 possible actions in the MDP for every state. The plane incurs a delay (cost) of 1 when it travels to a node without storm, and some higher delay (say 10) when it flies into a storm. For our numerical experiments, we consider different storm profiles, e.g. blocks of storm or randomly generated rows and columns of storms, to verify the versatility of our implementations, and to compare the nominal policy to the robust version over various uncertain grid maps.

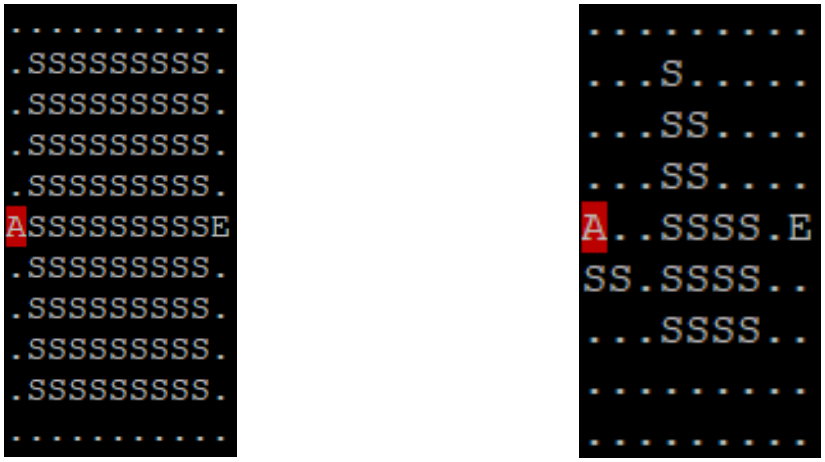
The nominal problem can be solved using the classic Bellman recursion. For the perturbed environment, we use robust MDP to find the path with the minimum delay under the worst possible storm profile. If the probability of storm is very low, then it is expected that the robust policy will incur a higher cost since it takes a longer detour to avoid the storm area as opposed to the nominal policy. However, if the environment has drastic fluctuations with higher storm probability, the nominal policy may sometimes fly through storms and incur a higher cost than the robust detour.

5.2. **Numerical Results.** In our experimental setup, we use the nominal environment to find the nominal policy. The robust policy is also computed on the nominal environment with additional information about the storm transition matrix. Then we use the gym Python package to simulate the policies in randomly perturbed environments. We run multiple iterations by varying β_i^a in 4.9 and 4.12, which is the level of uncertainty of the storm transition matrix. A larger β_i^a implies that the robust approach has to optimize over a larger distribution space.

5.2.1. *Checking the value function - Storm map with a standard shape.* We first evaluate the algorithms based on a standard storm map, as shown in Fig. 2a. In this experiment, we wish to demonstrate the effectiveness of our implemented robust algorithm by evaluating on a case where robust algorithm would get the privilege over the nominal case. More precisely, we try our best to set the empirical storm transformation to favour more good weather time-steps, while the storm would appear pretty rarely.

To simulate such environment, we set the probability of transformation to good weather to be 0.9, and storm to be 0.1. It is worth pointing out that under such circumstances, the transformation of the weather is independent from the weather of the current time step. We then severely perturb the transformation probabilities with 0.75 (so that the probability of maintaining a good weather decreases to 0.15, etc). We assign the cost of staying in a good weather location to be 1, while staying in a storm location to be 5.
We plot the comparison of the value functions for the robust and the nominal algorithm respectively. The values are represented by a number over each state. In this experiment with grid size of 11, there is only 1 storm, indicating the total number of states is $2^1 \times 11 \times 11$. In Fig. 3, we plot the values in the form of matrices, where the left two figures are the values from the nominal algorithm, and the right two figures are the values from the robust algorithm. The difference between the top two figures (Fig. 3a) and the bottom two figures (Fig. 3b) is that, the values in Fig. 3a are those states with the storm off, and the values in Fig. 3b are those states with the storm on.
We can observe from the values clearly that, the robust algorithm obtain a more pessimistic estimation of the cost for each locations (states). Even when there is no storm (Fig. 3a), the robust algorithm still indicates that the plane is very likely to be caught in a storm if it is inside the potential storm area. Our experimental results indicate the same as our analysis: if the initial state has no storm, the nominal algorithm would choose to go straight to the destination regardless of the huge potential the storm would occur, while the robust algorithm tends to be more conservative and takes the detour to the destination. On average of multiple roll-outs, the cost of the nominal algorithm in this case is 64.0, and of the robust algorithm is 20.0.

5.2.2. *Checking the cost - Storm map with a randomly generated shape.* The experiments in Sec. 5.2.1 provide a most friendly environment for the robust algorithm - the provided



(A) Storm map with a standard shape. The grid size is 11 by 11, with a square-shaped storm of size 9 by 9 in the middle. The experimental results on this map is demonstrated in Sec. 5.2.1.

(B) Storm map with a randomly generated shape. The grid size is 9 by 9, with 4 randomly generated rectangle-shaped storm. The experimental results on this map is demonstrated in Sec. 5.2.2.

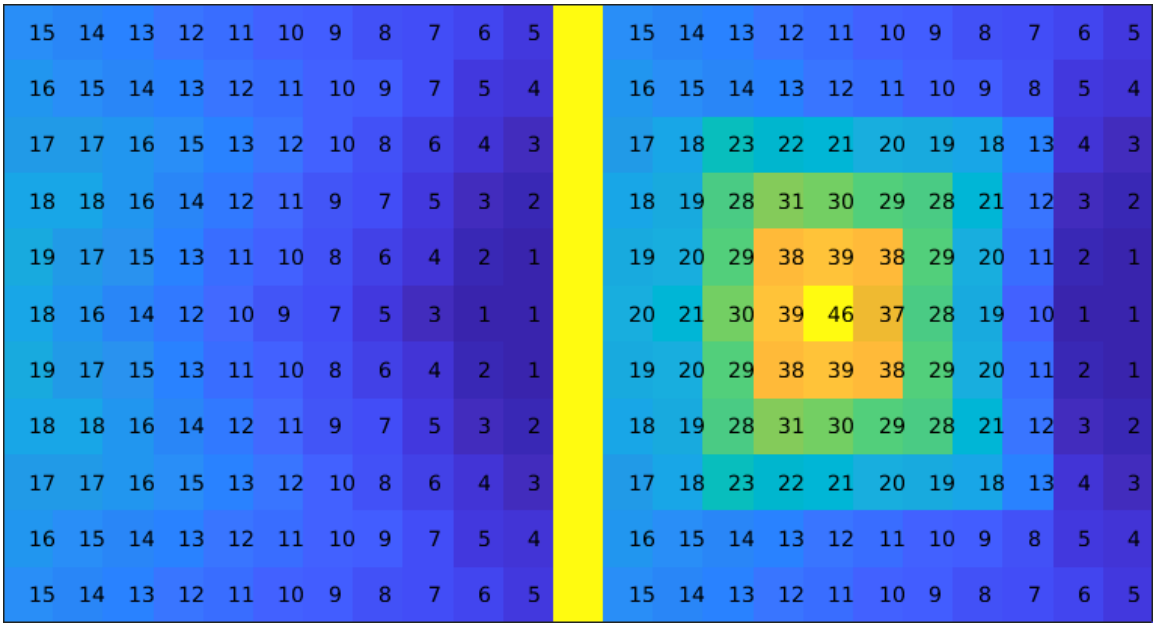
FIGURE 2. Two storm maps that are used in our experiments.

empirical transformation indicates much less probability that a storm would occur than the real environment. In this section, we compare the robust algorithm and the nominal algorithm over a randomly generated map (Fig. 2b), with 4 storms and grid size of 9, resulting a total number of $2^4 \times 9 \times 9$ states. We conduct a more detailed evaluation of the cost and perform the analysis.

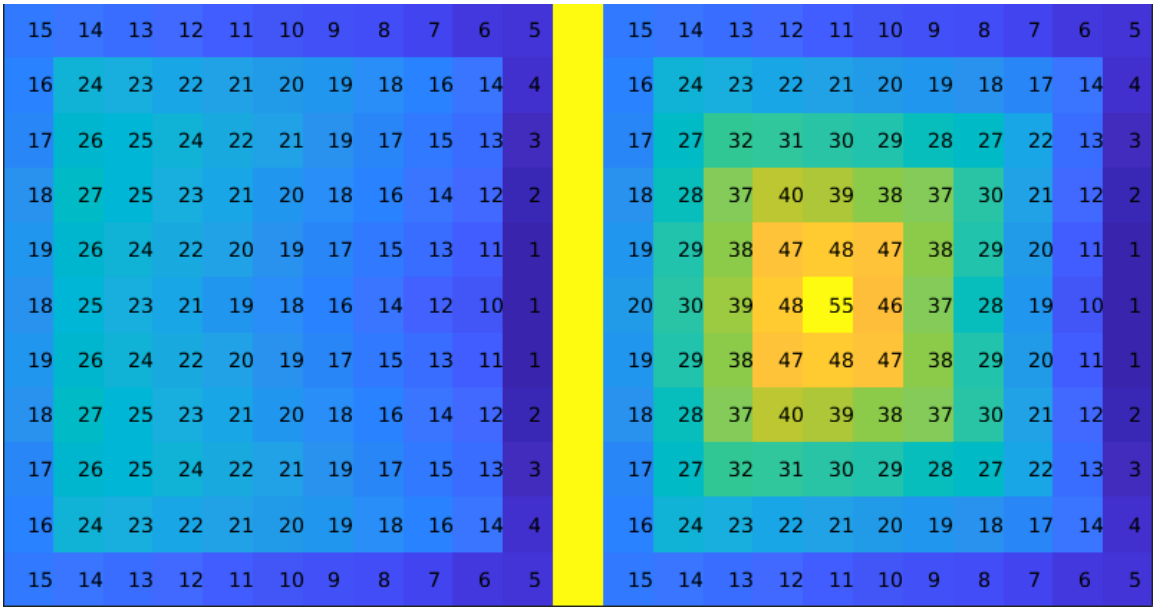
Two factors strongly affect the behaviour of the two algorithms: the perturbation factor ϵ (which is approximately proportional to β) and the cost over the storm place c (we assume the cost over the non-storm place is always 1).

In order to observe the behaviours over all the possible (ϵ, c) pairs, we start with the experiment to sweep both two parameters. As the cardinality of the states is significantly larger than the experiments in Sec. 5.2.2, we skip printing the value functions and instead provides the cost over all the possible (ϵ, c) pairs. For each (ϵ, c) pair, we run the roll-outs 30 times (to sufficiently take the randomness of the weather changes into account) and put them into a 2D matrix (Fig. 4). We refer readers to Fig. 4 for the detailed explanation. Consistent to our previous analysis, the robust algorithm performs better or equally good on almost every state compared to the nominal algorithm.
To further analyze the robust algorithm in detail, we further provide two error-bar curves in Fig. 5, with each fixing one parameter and sweeping the other. In Fig. 5a, we fix c to be 5 and sweep the ϵ, while in Fig. 5b, we fix ϵ to be 0.32 and sweep c. Once again, the robust algorithm outperforms the nominal one under almost any parameter settings.

Although in our algorithms, the robust algorithm is better under many different experimental settings, we do not claim the robust algorithm always provide us a better strategy under any case. For instance, if the true environment has higher probability of having good weather, then the overly conservative robust algorithm would instead provide us with a sub-optimal detour path, resulting in larger delays.



(A) Values calculated over the states where the storm is **off** (see the storm map in Fig. 2a). The left part contains the values from the nominal algorithm, while the right part contains the values from the robust algorithm. All the values are rounded to the nearest integer for the purpose of visualization.



(B) Values calculated over the states where the storm is **on** (see the storm map in Fig. 2a). The left part contains the values from the nominal algorithm, while the right part contains the values from the robust algorithm. All the values are rounded to the nearest integer for the purpose of visualization.

FIGURE 3. Values over all the $2^1 \times 11 \times 11$ states from the nominal value iteration (left) and the robust value iteration (right).

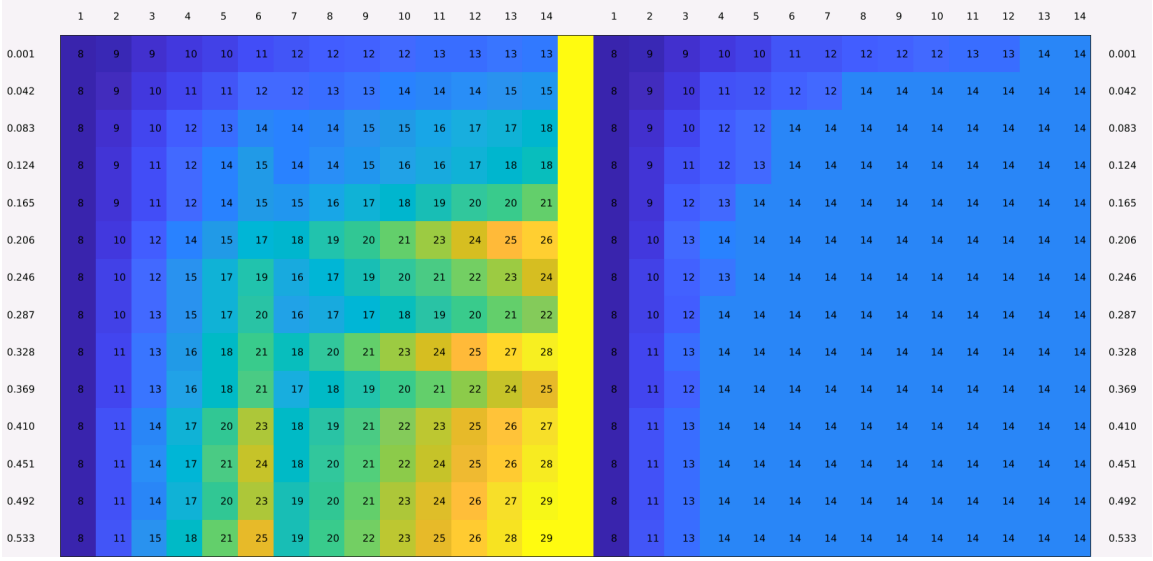


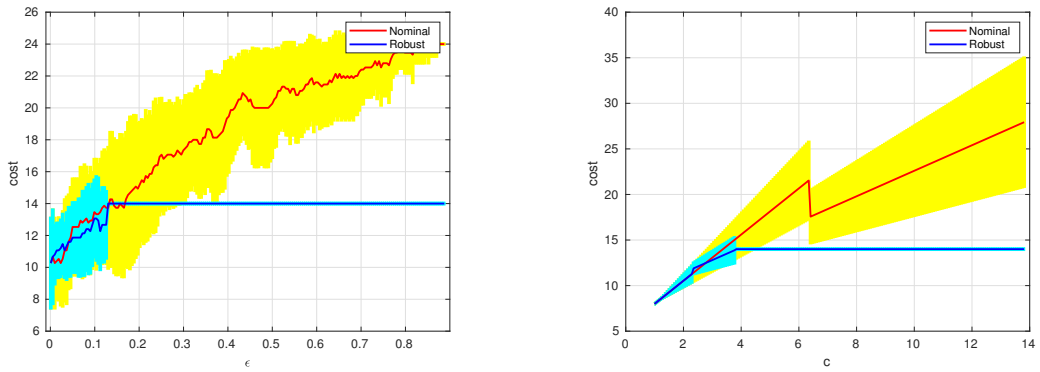
FIGURE 4. Average cost (smaller the better) when sweeping the parameter ϵ (perturbation to the environment, higher implies more significant) and c (the cost when caught in a storm for one time step. We assume staying in a non-storm place would have a fixed cost of 1). Each row is controlled by the same ϵ with varying c, and vice versa for each column. All the cost numbers are rounded to the nearest integer for the purpose of visualization. The left shows the costs from the nominal algorithm while the right from the robust algorithm. When $c = 1$, the cost is the same for staying in a storm or non-storm place so both algorithms would go straight to the destination with the cost always 8. The cost of 14 (for most of the sweeps in the robust algorithm) is obtained by going along a path that avoids any possibility of getting caught in a storm (inferred from Fig. 2b), and hence it is the upper bound for the cost in the robust algorithm. We can see that the robust algorithm always performs better or equally well compared to the nominal algorithm.

6. Application: Machine Replacement Problem

6.1. **Problem Formulation.** Machine replacement problems concerns the type of machine that deteriorates over years, and decisions need to be made over when to repair the machine. For simplicity, such problem can be modelled as a MDP process where the cost of repair is subjected to uncertainty. This formulation can be addressed by the robust type algorithms.

In reality, a factory may own high number of machines and they want a repair schedule that minimizes the expectation over all the machines with some probability confidence. Such formulation matches exactly with the chance constraint formulation.

So we investigated the performance of different formulation given two type of uncertainty set - uncertain reward and uncertain transition matrices. For the uncertain reward, we experimented the simple case that all the reward follows Gaussian distribution, so the problem can be solved via SOCP. An instance of this problem with fixed uncertainty in rewards, is shown in Fig 8, taken from [2].



(A) The average cost (vertical) and the standard deviation of the *epsilon*-sweep (horizontal). We fix c to be 5.

(B) The average cost (vertical) and the standard deviation of the c-sweep (horizontal). We fix ϵ to be 0.32.

FIGURE 5. The average cost and the standard deviation of the sweeps.

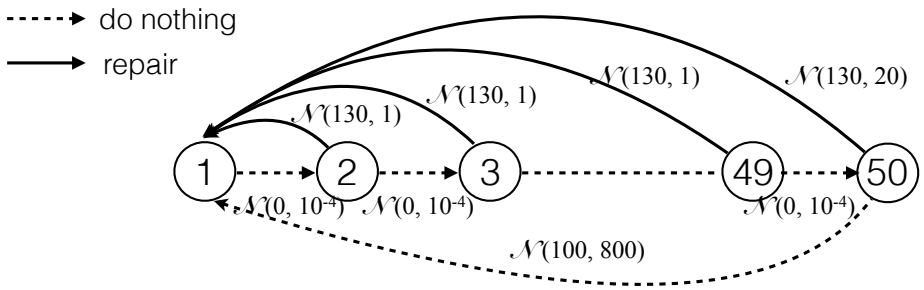


FIGURE 6. Machine replacement problem with uncertain rewards [2], each circle represents one state, the black arrow denotes the transition if the agent chooses to repair and the dotted arrow denotes the transition if the agent chooses to do nothing, each transition accompanies a uncertain reward from a Gaussian distribution.

For the second case, the uncertain transition matrix follows the Dirichlet distribution. The robust optimization is done using the likelihood and entropy models introduced in Section 3.

6.2. **Numerical Results.** For the uncertain Gaussian reward case, the cumulative reward histogram over different algorithms are displayed in Fig 7. We ran 1000 trials for each experiment. Each experiment uses an environment with deterministic reward sampled from a random distribution.

For the uncertain transition matrix case, the cumulative cost for nominal and robust policies are shown in Fig 9. It is seen that the nominal case is actually better than the robust version in our experimental settings. This is probably because the environment was more favourable for nominal and did not have sufficient perturbations, so the conservative robust policy is sub-optimal. The paper [2] mentions that chance constraint policy once again provides the optimal policy for this problem setting.

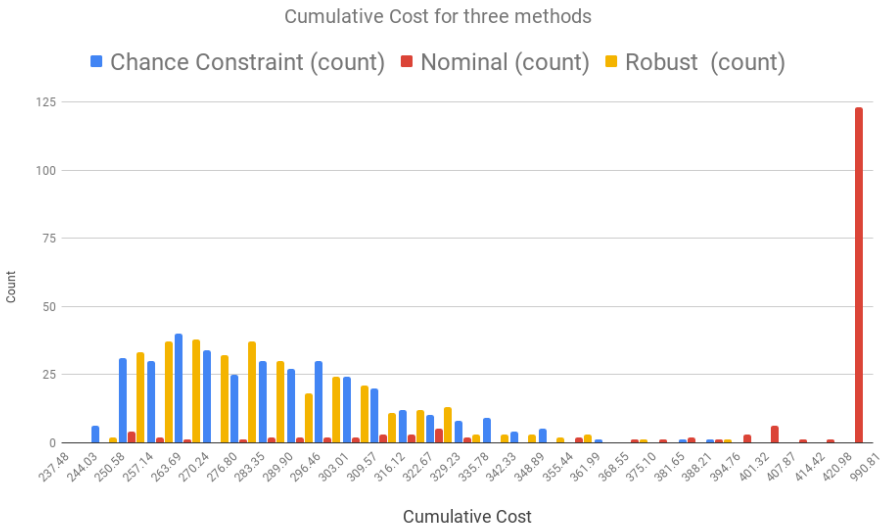


FIGURE 7. Cumulative cost of Chance constraint, Nominal, and robust

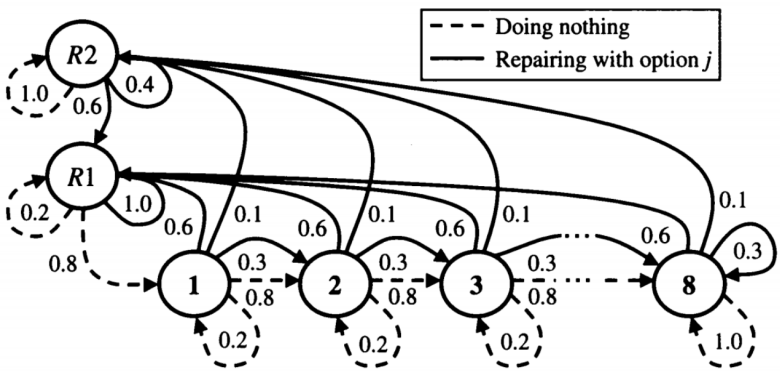


FIGURE 8. Machine replacement problem with transitions, the figure described the problem is from [2], each circle represents one state, the black arrow denotes the transition if the agent chooses to repair and the dotted arrow denotes the transition if the agent chooses to do nothing, each action has multiple possible outcomes with a probability on the line.

7. CONCLUSION

We verified the expected behavior of nominal, robust and chance constrained approach to Markov Decision Processes:

- **Nominal strategy**: This is 'blind to risk', so it incurs a high cost or low reward if the environment has large fluctuations.
- **Robust strategy**: This is conservative, so it performs better than the nominal only when the environment has drastic variations. However, it does not use prior

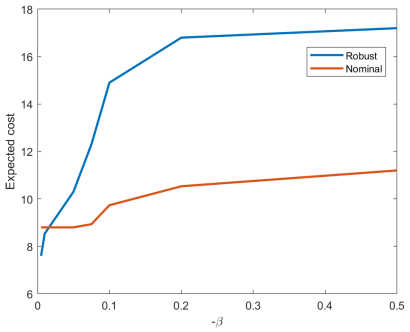


FIGURE 9. Expected cost of machine replacement problem with uncertain transition matrix for nominal and robust policies

knowledge of the perturbation statistics but optimizes for the worst case scenario, often resulting in a sub-optimal policy.

- **Chance constrained policy**: This is a compromise between the 2 previous strategies. i.e. it uses extra information about environment statistics to calculate a policy based on a preferred level of risk. Hence it outperforms the robust strategy when compared over multiple iterations.

REFERENCES

[1] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust optimization*, volume 28. Princeton University Press, 2009.

[2] Erick Delage and Shie Mannor. Percentile optimization for markov decision processes with parameter uncertainty. *Operations research*, 58(1):203–213, 2010.

[3] Erich L Lehmann and Joseph P Romano. *Testing statistical hypotheses*. Springer Science & Business Media, 2006.

[4] Arnab Nilim and Laurent El Ghaoui. Robust control of markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005.

[5] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[6] Wolfram Wiesemann, Daniel Kuhn, and Berç Rustem. Robust markov decision processes. *Mathematics of Operations Research*, 38(1):153–183, 2013.

[7] Huan Xu and Shie Mannor. Distributionally robust markov decision processes. In *Advances in Neural Information Processing Systems*, pages 2505–2513, 2010.