International Journal of Automation and Computing X(X), X X, X-X
DOI: XXX

Solving Markov Decision Processes with Downside Risk Adjustment

Abhijit Gosavi* Anish Parulekar**
*219 Engineering Management Building, Dept. of Engineering Management and Systems Engineering
Missouri University of Science and Technology, Rolla, MO 65409, USA
** Axis Bank, Mumbai, India

Abstract: Markov decision processes (MDPs) and their variants are widely studied in the theory of controls for stochastic discrete-event systems driven by Markov chains. Much of the literature focusses on the risk-neutral criterion in which the expected rewards, either average or discounted, are maximized. There exists some literature on MDPs that takes risks into account. Much of this addresses the exponential utility (EU) function and mechanisms to penalize different forms of variance of the rewards. EU functions have some numerical deficiencies, while variance measures variability both above and below the mean rewards; the variability above mean rewards is usually beneficial and should not be penalized/avoided. As such, risk metrics that account for pre-specified targets (thresholds) for rewards have been considered in the literature, where the goal is to penalize the risks of revenues falling below those targets. Existing work on MDPs that takes targets into account seeks to minimize risks of this nature. Minimizing risks can lead to poor solutions where the risk is zero or near zero, but the average rewards are also rather low. In this paper, hence, we study a risk-averse criterion, in particular the so-called downside risk, which equals the probability of the revenues falling below a given target, where, in contrast to minimizing such risks, we only reduce this risk at the cost of slightly lowered average rewards. A solution where the risk is low and the average reward is quite high, although not at its maximum attainable value, is very attractive in practice. To be more specific, in our formulation, the objective function is the expected value of the rewards minus a scalar times the downside risk. In this setting, we analyze the infinite horizon MDP, the finite horizon MDP, and the infinite horizon semi-MDP (SMDP). We develop dynamic programming and reinforcement learning algorithms for the finite and infinite horizon. The algorithms are tested in numerical studies and show encouraging performance.

Keywords: downside risk; Markov decision processes; reinforcement learning; dynamic programming; targets; thresholds

1 Introduction

Markov decision processes (MDPs) have been studied widely in the literature for classical objective functions, such as discounted reward over an infinite/finite horizon, average reward over an infinite horizon, and total reward over a finite time horizon. MDPs can be solved via dynamic programming (DP) when the underlying transition probability model is available. When the transition probability models are not available, either because the underlying transition dynamics are too complex or because the state-action space is too large, MDPs can be solved within simulators via a methodology called reinforcement learning (RL) [1, 2]. Recently, RL methods have been used in a variety of problems ranging from predicting aircraft taxi-out times [3] through supply chain management [4] to helicopter control [5].

The classical objectives, unfortunately, are unable to capture risk considerations. In other words, classical objective functions disregard the issue of reducing *variability* in the returns (rewards/revenues), although variability is often undesirable. In this paper, we are interested in *reducing* variability below a given (known) target, or threshold, for the revenues. The specific form of variability that we are interested in is called downside risk, i.e., the probability of the revenues falling below the target specified, or equivalently, the probability of the costs exceeding the target specified (in case of costs, this is often called the expected shortfall). Further, in this paper, we will use a risk-adjusted (also called risk-penalized in the literature) objective function of the following format that is *maximized*:

$$\mathsf{E}[X] - \theta \mathsf{Risk}[X], \tag{1}$$

where θ is a positive risk-averseness coefficient, X denotes the rewards or returns, while $\mathsf{E}[.]$ and $\mathsf{Risk}[.]$ denote the expectation and risk operators respectively. The risk operator could be variance, semi-variance, downside risk etc. The format used in Equation (1) will be called the risk-adjusted format because it equals the expected rewards minus a positive scalar, θ, times the risk. Before delving into details of how the above format can be employed within MDPs, we present a simple motivating example from risk economics. Consider the situation in which you have two choices:

- Scenario 1: Make \$800 with a probability of 0.9 and \$100 with a probability of 0.1.
- Scenario 2: Make \$800 with a probability of 0.8 and \$450 with a probability of 0.2.

Note that both scenarios result in the same *expected* returns of \$730. A decision-maker who likes to take risks will select Scenario 1 because the chances of obtaining the highest reward are higher with it. However, a risk-averse person is likely to choose Scenario 2 because it "seems" less risky. For instance, with Scenario 2, in the worst case, one makes \$450, which is higher than \$100 that one makes with Scenario 1 in the worst case. So how does one distinguish

Manuscript received date; revised date

between the two if one is risk-averse? In the setting of artificial intelligence that we study in this paper, it is necessary to come up with an approach that will crunch numbers and automatically select the risk-averse solution for the agent. One approach to computing risk is to set a reasonable target, and then compute the downside risk, i.e., the probability of revenues falling below that target. Let us assume that the target is set to \$100, the lowest amount achievable from either scenarios. Then, the downside risk for Scenario 1 is 0.1 and that for 2 is 0. Hence, using the objective function format in Equation (1) will lead to a higher score for Scenario 2, and the decision-maker will hence choose Scenario 2. In this paper, we will consider an objective function of this nature for solving MDPs. The motivation is to derive solutions that have slightly lowered average revenues but significantly lower risks.

A body of literature exists on curtailing risks in the rewards (or costs) in MDPs of which we now present a brief review. A major chunk of this is devoted to using the exponential utility (EU) function [6, 7, 8], which, when used in the MDP context, leads to the so-called risk-sensitive objective function. On the other hand, Filar *et al.* [9] develop the *variance*-adjusted (VA) framework. See also [10] for what is possibly the first definition of variance within MDPs. However, we do not find any Bellman equation in these works. A so-called policy-gradient algorithm based on a VA-based Bellman equation can be found in Sato and Kobayashi [11]. The algorithm in Gosavi [12] uses one-step variance as a risk measure for the infinite horizon and proposes a policy iteration algorithm for dynamic programming, while the algorithms in [13, 14] are for long-run variance.

Outside the EU function and variance, other metrics have been studied in conjunction with MDPs. Mihatsch and Neuneier [15] introduce a scaling parameter that transforms the reward function in MDPs. Geibel [16] introduces the notion of "risky states" that should be avoided, while Heger [17] studies what is called worst-case risk. A subset of risk-related notions have also stoked an interest in management problems [18, 19].

In this paper, our goal is to consider a threshold/target for revenues/rewards/returns and ensure that the probability of the revenues falling below the threshold is reduced (this is the same as reducing the probability of costs rising *above* a given threshold). There is some literature on MDPs that considers targets: First, there is work aimed at *minimizing* the probability that the total discounted reward falls below a target [20, 21, 22], as opposed to considering a risk-adjusted objective function in an un-discounted setting that we consider here. Second, a so-called *semi-variance* from a target has also been used as a risk measure within a risk-adjusted objective function [23]; semi-variance measures the square of the absolute deviation below the target.

There are numerical advantages to using downside risk over other risk metrics. The EU function has a number of drawbacks that have been discussed in [24]. In particular, the most important disadvantage of the EU function is that the associated RL algorithm can break down numerically in problems where there is large variability in rewards in different states. Variance, as stated above, is another popular risk metric, but unfortunately, it computes variability both above *and* below the mean. Usually, variability above the mean rewards is useful, and should not be penalized, which is precisely what a so-called variance-penalized objective leads to. Hence, using a specific target for the rewards, as done within downside risk, is more appropriate for measuring risk than the average reward. Finally, as stated above, semi-variance-adjusted average rewards, which do account for targets, have been considered in the literature [23], but it needs to be noted that the numerical value of the target-semi-variance can at times become very large — causing computational issues in RL algorithms. The downside risk, on the other hand, is a probability, and hence its value never becomes too large — thereby offering a numerical advantage over target semi-variance.

Contributions of this paper: This paper seeks to present, to the best of our knowledge, for the first time, the downside risk criterion in a risk-adjusted format for the MDP, SMDP, and the finite horizon MDP. The paper also provides the underlying theory and algorithms, via DP and RL, for solving these problems. Finally, numerical results are presented to reinforce the usefulness of the framework proposed. The framework holds key advantages over other risk metrics, which were discussed above.

The rest of this paper is organized as follows. We first consider the infinite time horizon in Section 2 within which we cover the MDP and the SMDP. Section 3 provides the theory for the finite time horizon problem. Numerical results are provided in Section 4, while conclusions drawn from this research are presented in the final section.

2 Infinite Time Horizon

We first consider the case of the infinite time horizon for the SMDP, whose algorithms often have straightforward extensions to MDPs, since the latter are a special case of the former. See [25, 26] for the necessary background on the theory underlying MDPs/SMDPs. We will first present some preliminaries, then the underlying theory, relevant to downside risk, for MDPs and finally the same for SMDPs and the finite horizon.

2.1 Preliminaries

We will be using the following notation. $\mathcal{S}$ will denote the set of states in the MDP/SMDP, while $\mathcal{A}(i)$ will denote the set of actions in state i. Let $r(i, a, j)$ denote the reward earned in going from state i to state j under action a. Also, let $p(i, a, j)$ and $t(i, a, j)$ denote the probability and time respectively — associated with the same transition. We will use μ to denote a policy for which $\mu(i)$ will denote the (deterministic) action to be chosen in state i. Also, $\mathbf{P}_\mu$ and $\mathbf{R}_\mu$ will denote the transition probability and transition reward matrices, respectively, associated with policy μ. Also, note that we will use the following notation for the expected immediate reward and time: $\bar{r}(i, a) = \sum_{j \in \mathcal{S}} p(i, a, j) r(i, a, j)$ will denote the expected immediate reward in state i when action a is chosen in it, while $\bar{t}(i, a) = \sum_{j \in \mathcal{S}} p(i, a, j) t(i, a, j)$ will denote the expected immediate transition time out of state i when action a is chosen in it. Further, we will use τ to denote the target, which will be in terms of dollars for MDPs and dollars per unit time for SMDPs. We now make an important assump-

tion about the nature of MDPs/SMDPs that we will study here.

Assumption 1 *The Markov chain underlying every policy μ in the infinite horizon MDP/SMDP is regular, i.e., the transition probability matrix can be raised to a sufficiently large finite-valued power such that each element in the matrix is non-zero.*

We now define the average reward and the downside risk.

Definition 1 *For a given deterministic, stationary policy μ, the average reward in an infinite horizon SMDP is defined as:*

$$\rho_\mu = \frac{\sum_{i\in\mathcal{S}} \Pi^\mu(i) \sum_{j\in\mathcal{S}} p(i,\mu(i),j) r(i,\mu(i),j)}{\sum_{i\in\mathcal{S}} \Pi^\mu(i) \sum_{j\in\mathcal{S}} p(i,\mu(i),j) t(i,\mu(i),j)}, \quad (2)$$

where $\Pi^\mu(i)$ denotes the steady-state probability (invariant probability) for state $i \in \mathcal{S}$ of the Markov chain underlying policy μ.

Definition 2 *For a given deterministic, stationary policy μ, the downside risk in an SMDP is defined as: $DR^\mu =$*

$$\sum_{i\in\mathcal{S}} \Pi^\mu(i) \sum_{j\in\mathcal{S}} p(i,\mu(i),j) I(r(i,\mu(i),j) < \tau t(i,\mu(i),j)) \quad (3)$$

where $I(.)$ denotes the indicator function (which equals 1 if the condition inside the brackets is true and 0 otherwise).

We can obtain the counterparts of MDPs for both of the above definitions by identically setting the time $t(.,.,.) \equiv 1$ in each.

We can now define our objective function for a given policy μ using the format in Equation (1) as follows:

$$\phi_\mu \equiv \rho_\mu - \theta DR^\mu, \quad (4)$$

where in the above ρ_μ denotes the average reward of the policy and DR^μ denotes the downside risk of using policy μ. The above expression in Equation (4) will be referred to as the downside-risk-adjusted (DRA) score. It is important to note here that the positive scalar θ is proportional to the risk-averseness of the decision-maker; the greater the degree of risk-averseness, the higher should be this value. In other words, a decision-maker who has no averseness to the downside risk should use a value of 0 for θ, while the decision-maker with a high degree of risk-averseness should use a larger value. In general, however, very large values of θ can cause the above DRA score to be too biased to the downside risk. Thus, this is a parameter that must be set at a suitable value after taking into account the tolerance level of the decision-maker towards downside risk. It also needs to be highlighted that a high value of θ will produce low downside risk, but this will occur at the cost of reducing the average reward, ρ_μ. Therefore in practice, the actual value of θ has to be determined depending on not only the risk tolerance level of the decision-maker but also on the experimental set up and how it influences the average reward and the risk. We discuss examples in the context of numerical results later in the paper.

We now present the theory for SMDPs.

Bellman equations: To develop policy and value iterations of DP for the objective function at hand, we present the Bellman equations for the SMDP from which these algorithms will follow. We will now define the following function that will be used occasionally in the remainder of the paper to make our presentation compact:

$$w_\tau(i,a,j) = r(i,a,j) - \theta I(r(i,a,j) < \tau t(i,a,j)) \quad (5)$$

for all $i \in \mathcal{S}$, $j \in \mathcal{S}$, and $a \in \mathcal{A}(i)$. Further, we define $\bar{w}_\tau(i,a) = \sum_{j\in\mathcal{S}} p(i,a,j) w_\tau(i,a,j)$ for all i,j in $\mathcal{S}$ and every $a \in \mathcal{A}(i)$. When this function is used in the context of MDPs, $t(i,a,j) = 1$ for all $i \in \mathcal{S}$, $j \in \mathcal{S}$, and $a \in \mathcal{A}(i)$.

Theorem 1 *(i) (Bellman equation for a given policy) If a scalar $\phi \in \Re$ and an $|S|$-dimensional finite vector $\vec{h}$ satisfy for all $i \in \mathcal{S}$:*

$$\phi \bar{t}(i,\mu(i)) + h(i) = \sum_{j\in\mathcal{S}} p(i,\mu(i),j) \left[w_\tau(i,\mu(i),j) + h(j) \right],$$

then ϕ is the DRA score associated with the policy μ.
(ii) (Bellman optimality equation) Assume that a scalar ϕ^* and an $|\mathcal{S}|$-dimensional finite vector $J(i)$ satisfy for all $i \in \mathcal{S}$

$$J(i) = \max_{a\in\mathcal{A}(i)} \left[\sum_{j\in\mathcal{S}} p(i,a,j) [w_\tau(i,a,j) - \phi^* t(i,a,j) + J(j)] \right]. \quad (6)$$

Any policy that attains the max in the right-hand side of the above will be an optimal policy, i.e., it will generate the maximum value for the DRA score.

These results are extensions of the classical results for the risk-neutral case [26], and hence we skip the proofs. The proofs can be easily obtained by replacing $r(.,.,.)$ in the proof for the classical risk-neutral case by the function $w(.,.,.)$,

2.2 Dynamic Programming

For the infinite time horizon, we present the so-called policy iteration algorithm which works for SMDPs as well as MDPs.

Step 1. Set k, the number of iterations, to 1. Select any arbitrary policy and denote it by $\hat{\mu}_k$, while $\hat{\mu}^*$ will denote the optimal policy.

Step 2. (Policy Evaluation) Solve the following linear system of equations:

$$h^k(i) = \sum_{j=1}^{|\mathcal{S}|} p(i,\mu_k(i),j) \left[w_\tau(i,\mu_k(i),j) - \phi^k t(i,\mu_k(i),j) + h^k(j) \right].$$

In the above, the unknowns are the h^k terms as well as ϕ^k. Any one of the h^k terms should be set to 0 in order to obtain a solution.

Step 3. (Policy Improvement) Choose a new policy $\hat{\mu}_{k+1}$ so that for all $i \in \mathcal{S}$

$$\mu_{k+1}(i) \in$$

$$\arg\max_{a\in\mathcal{A}(i)}\left[\sum_{j=1}^{|\mathcal{S}|}p(i,a,j)(w_\tau(i,a,j)-\phi^k t(i,a,j)+h^k(j))\right].$$

The action selection should be performed in a manner such that when possible, one should set $\hat{\mu}_{k+1}=\hat{\mu}_k$; this is done to avoid cycling.

Step 4. If the new policy is identical to the old one, i.e., if $\mu_{k+1}(i)=\mu_k(i)$ for each $i\in\mathcal{S}$, then set $\mu^*(i)=\mu_k(i)$ for every $i\in\mathcal{S}$ and stop. Otherwise, increase k by 1, and return to Step 2.

The convergence of the algorithm follows directly from that of the classical risk-neutral algorithm, via replacement of $r(.,.,.)$ by $w(.,.,.)$, and hence is not presented in detail. A relative value iteration algorithm using the DRA function follows directly from the Bellman equations proposed above for the MDP. Its convergence will also follow along lines of the same for the classical risk-neutral case [25]; details of the algorithm are provided in Appendix A1 for the sake of completeness.

2.3 Reinforcement Learning

In this section, we develop an RL algorithm for infinite horizon SMDPs. In RL, the goal is to solve the problem either in a simulator or in the actual system via trial and error, typically under the assumption that the transition probabilities needed in DP are not available. In other words, the algorithm must update its values after each transition in the system (or simulator). RL algorithms, hence, have the potential to solve complex problems whose transition probabilities are hard to find.

We develop an algorithm along the lines of [27, 23], where the objective function is estimated simultaneously with the updating of the value function. Our proposed algorithm will work for MDPs by setting $t(.,.,.)\equiv 1$ everywhere. Instead of solving the Bellman equation defined in Equation (6), our algorithm will seek to solve the following equation: For all $i\in\mathcal{S}$ and any scalar $\eta\in(0,1)$:

$$J(i)=$$

$$\max_{a\in\mathcal{A}(i)}\left[\sum_{j\in\mathcal{S}}p(i,a,j)[w_\tau(i,a,j)-\phi t(i,a,j)+\eta J(j)]\right]. \quad (7)$$

In the above, η is an artificially introduced positive scalar, which is close to 1, e.g., $\theta=0.99$. The motivation for introducing it is that while it helps the underlying Bellman equation (i.e., Equation (7)) to approximate the actual Bellman equation, it also leads to a unique solution for the underlying Bellman equation and generates a convergent algorithm. In the literature, such a scalar has been also used in policy gradient algorithms to force a unique solution [28]. It is essentially a tuning parameter whose value must be chosen carefully. We discuss this in more detail below in the discussion that follows the RL algorithm.

We now make the following assumption under which solving the above equation will yield the same solution as solving the actual Bellman equation. We will comment on the verifiability of this assumption later in the section where we present numerical results.

Assumption 2 *There exists a value for $\bar{\eta}$ in the interval $(0,1)$ such that for all $\eta\in(\bar{\eta},1)$, the unique solution, J, of Equation (7) with ϕ set to equal ϕ^* produces a policy d defined as follows for all $i\in\mathcal{S}$*

$$d(i)\in\arg\max_{a\in\mathcal{S}}\left[\bar{w}_\tau(i,a)-\phi^*\bar{t}(i,a)+\eta\sum_j p(i,a,j)J(j)\right]$$

whose DRA score equals ϕ^*.

Note that under the assumption above, when η lies in $(0,\bar{\eta})$, the unique solution of Equation (7) with $\phi=\phi^*$ will be the same as the solution of the Bellman optimality equation. We now define a so-called Q-factor that is used in algorithms of the Q-Learning type. For all (i,a):

$$Q(i,a)=\sum_{j\in\mathcal{S}}p(i,a,j)\left[w_\tau(i,a,j)-\phi t(i,a,j)+\eta J(j)\right], \quad (8)$$

where $J(.)$ denotes the unique solution of Equation (7). Equations (7) and (8) imply that for every $i\in\mathcal{S}$, $J(i)=\max_{a\in\mathcal{A}(i)}Q(i,a)$, which from (8) implies that $Q(i,a)=$

$$\sum_{j\in\mathcal{S}}p(i,a,j)\left[w_\tau(i,a,j)-\phi t(i,a,j)+\eta\max_{b\in\mathcal{A}(j)}Q(j,b)\right]. \quad (9)$$

This motivates the following RL algorithm (along the lines of the semi-variance algorithm [23]):

$$Q^{k+1}(i,a)\leftarrow(1-\alpha^k)Q^k(i,a)+$$

$$\alpha^k\left[w_\tau(i,a,j)-\phi^k t(i,a,j)+\eta\max_{b\in\mathcal{A}(j)}Q^k(j,b)\right],$$

where ϕ^k is updated simultaneously to its optimal value.

Algorithm Steps: In the algorithm below, we will use what is called an ϵ-greedy selection in the literature [1]. In an ϵ-greedy strategy, actions are selected in a manner such that all actions have the same probability of getting selected in the first iteration, but the probability of selecting the non-greedy action is gradually reduced with every iteration.

Initialization: Set k, the number of iterations, to 0. Set for all (i,a), where $i\in\mathcal{S}$ and $a\in\mathcal{A}(i)$, $Q^k(i,a)\leftarrow 0$. Set ϕ^k, the estimate of the DRA score in kth iteration to 0. Also, set the scalars, TW^k, which measures the total accumulated value of the risk-adjusted immediate reward $(w_\tau(.,.,.))$ under greedy actions, to 0 and TT^k, which measures the total time spent in greedy actions, to a small positive value, e.g., 0.01. Set η to a value close to 1, e.g., 0.99. Let α^k and β^k be step-sizes that are decayed according to standard RL rules. Set $k_{\max}$, the number of iterations for which the algorithm is run, to a large enough integer. Initiate system simulation at any arbitrary state.

Repeat the following steps in a sequence until $k=k_{\max}$:

• Let the current state be i. An action u will be considered greedy if $u=\arg\max_{b\in\mathcal{A}(i)}Q^k(i,b)$. Select action a using an ϵ-greedy strategy. In case there is a tie in finding the greedy action, the tie is broken randomly.

• Simulate action a. Let the next state be j. Update $Q(i,a)$ as follows:

$$Q^{k+1}(i,a)\leftarrow(1-\alpha^k)Q^k(i,a)+$$

$$\alpha^k[w_\tau(i,a,j) - \phi^k t(i,a,j) + \eta \max_{b\in\mathcal{A}(j)} Q^k(j,b)].$$

• If a is greedy, update ϕ, TW and TT using the following:

$$\phi^{k+1} \leftarrow (1-\beta^k)\phi^k + \beta^k \frac{TW^k}{TT^k};$$

$$TW^{k+1} \leftarrow TW^k + w_\tau(i,a,j); TT^{k+1} \leftarrow TT^k + t(i,a,j).$$

• Increment k by 1. If $k < k_{\max}$, set $i \leftarrow j$.

Termination: For each $i \in \mathcal{S}$, select $d(i) \in \arg\max_{b\in\mathcal{A}(i)} Q^k(i,b)$. The policy returned is d. Stop.

In the above, the step sizes, α^k and β^k, are decayed according to a rule such as $A/(B+k)$, where A and B are suitable positive scalars. In general, every step size in RL will be assumed to satisfy the following condition:

Assumption 3

$$\sum_{k=1}^{\infty} \alpha^k = \infty; \quad \sum_{k=1}^{\infty} (\alpha^k)^2 < \infty.$$

Setting the value for η: The value of η must be set as close to 1 as is possible. Our experiments suggested that a value of 0.99 is often sufficient in practice to obtain the optimal solution. However, on large-scale problems, where the optimal is unknown and one typically benchmarks against heuristics, one may have to use trial and error to generate behavior superior to that of the heuristics. As a rule of thumb, one can start from $\theta = 0.99$ and use increasing values until the desired behavior is obtained. Of course, as stated above, the value should always be less than 1.

The proof of convergence of this algorithm is a straightforward extension of an existing result; hence, we provide the result in Appendix A2.

3 Finite Time Horizon

We now present an analysis of the finite horizon case. In the finite horizon problem, our goal will be to maximize the expected sum of the risk-adjusted rewards earned over the finite time horizon. It turns out that under certain assumptions, the finite horizon problem can be studied as a special case of the stochastic shortest-path problem (SSP) [2]. In this section, we first present some preliminaries regarding the finite horizon MDP, then a forward DP algorithm, and finally an RL algorithm.

3.1 Preliminaries

We begin with some assumptions that we make about the finite horizon MDP, and then present some notation that we will need, before defining the objective function.

Assumption 4 *Every policy possible in the problem is proper, i.e., under every policy the system inevitably reaches a terminal state, which is unique, with a positive probability after a finite number of transitions.*

Assumption 5 *The starting state of the system is known and fixed.*

Assumption 6 *The terminal (ending) state of the system is unique and absorbing, and it generates no rewards.*

In the finite horizon problem, the state of the infinite horizon MDP is replaced by a state-stage pair. Thus, (i,s) will denote the pair of state i and stage s. If μ denotes a policy, $\mu(i,s)$ will denote the action selected in the state-stage pair (i,s). Also, $r(i,s,a,j,s+1)$ will denote the immediate reward earned in going from state-stage pair (i,s) to state-stage pair $(j,s+1)$; similarly, $p(i,s,a,j,s+1)$ will denote the transition probability under the same transition. Our risk-adjusted immediate reward can then be defined as:

$$w(i,s,a,j,s+1) = r(i,s,a,j,s+1) - \theta \times I(r(i,s,a,j,s+1) < \tau), \tag{10}$$

where $I(.)$ is the indicator function as defined above. This allows us to define the expected value of the risk-adjusted immediate reward as follows:

$$\bar{w}(i,s,a) = \sum_{j\in\mathcal{S}} p(i,s,a,j,s+1) w(i,s,a,j,s+1). \tag{11}$$

Using the above definitions, and noting that N will denote the finite number of stages in this setting, we now define the following function:

Definition 3 *The long-run expected value of the total risk-adjusted reward earned by a policy μ starting at state i in stage 1 over N stages is:*

$$\phi_\mu(i,1) \equiv \mathsf{E}_\mu \left[\sum_{s=1}^{N} \bar{w}(x_s, s, \mu(x_s,s)) | x_1 = i \right],$$

where x_s denotes the state occupied by the system in the sth stage.

The starting state in the finite horizon MDP will be unique via Assumption 4. And our objective function will be $\phi_\mu(1,1)$ where the initial state will be numbered 1. We note that $J(i,s)$ will be used to denote the element of the value function for state i in stage s of the finite horizon MDP. The associated Bellman optimality equation, which follows from the same for the risk-neutral case (see [25]) is provided below.

Theorem 2 *Under Assumptions 4-6 for a finite horizon MDP, there exists a unique solution for the following equation: For every $i \in \mathcal{S}$ and $s = 1,2,\ldots,N$,*

$$J^*(i,s) = \max_{a\in\mathcal{A}(i,s)} \left[\sum_{j\in\mathcal{S}} p(i,s,a,j,s+1) \left[w(i,s,a,j,s+1) + J^*(j,s+1) \right] \right]. \tag{12}$$

3.2 DP Algorithm

We are now ready to present the DP algorithm, based on value iteration, that solves the Bellman equation presented above, and follows from it.

Step 1. Select an $\epsilon > 0$. Set $k = 0$. Set arbitrary values, e.g., 0, to $J^0(i,t)$ for all $i \in \mathcal{S}$ and $t = 1,2,\ldots,N+1$. Set $s = 1$. Let i_* be the starting state.
Step 2. Perform the following update for all $i \in \mathcal{S}$, except when $s = 1$ in which case perform it only for i_*.

$$J^{k+1}(i,s) =$$

$$\max_{a\in\mathcal{A}(i,s)}\left[\sum_{j\in\mathcal{S}} p(i,s,a,j,s+1)\left[w(i,s,a,j,s+1)+J^k(j,s+1)\right]\right].$$

Step 3. Increment s by 1. If $s = N+1$ go to Step 4. Else, return to Step 2.
Step 4. Check if $||\vec{J}^{k+1} - \vec{J}^k||_\infty < \epsilon$. If true, go to Step 5. If this is not true, set $s = 1$, increase k by 1, and return to Step 2.
Step 5. The optimal action in each state-stage pair, (i, s), is determined as follows:

$$\arg\max_{a\in\mathcal{A}(i,s)}\left[\sum_{j\in\mathcal{S}} p(i,s,a,j,s+1)\left[w(i,s,a,j,s+1)+J^k(j,s+1)\right]\right].$$

Assumption 6 requires that the terminal state in stage $(N+1)$ be reward-free. This is ensured by the fact that $J(i, N+1)$ is never updated for any i, and thus always remains at zero. The proof of convergence of the algorithm is provided in Appendix A3.

3.3 RL Algorithm

The result for the value function used in DP, i.e., Theorem 2, can be extended to Q-values. To that end, we first define the Q-factor in terms of $J^*(.,.)$, which was defined above via Theorem 2, as follows: For every $i \in \mathcal{S}$, $s = 1, 2, \ldots, N$, and $a \in \mathcal{A}(i, s), Q^*(i, a, s) =$

$$\sum_{j\in\mathcal{S}} p(i,s,a,j,s+1)\left[w(i,s,a,j,s+1)+J^*(j,s+1)\right],$$

which implies that for every $i \in \mathcal{S}$ and $s = 1, 2, \ldots, N$,

$$J^*(i,s) = \max_{b\in\mathcal{A}(j,s)} Q^*(j,b,s).$$

Then, using the above, Theorem 2 can be written in terms of the Q-values as follows:

Theorem 3 *Under Assumptions 4-6 for a finite horizon MDP, there exists a unique solution for the following equation: For every $i \in \mathcal{S}$, $s = 1, 2, \ldots, N$, and $a \in \mathcal{A}(i, s)$*

$$Q^*(i,a,s) =$$

$$\sum_{j\in\mathcal{S}} p(i,s,a,j,s+1)\left[w(i,s,a,j,s+1)+\max_{b\in\mathcal{A}(j,s+1)} Q^*(j,b,s+1)\right]. \quad (13)$$

The above suggests an RL algorithm that can be used for the finite horizon problem when Assumptions 4–6 hold. We now present details of the algorithm.

Initialization For all (i, u), where $i \in \mathcal{S}$, $s = 1, 2, \ldots, N+1$, and $u \in \mathcal{A}(i, s)$, set $Q(i, s, u) = 0$. Set k, the number of state changes, to 0. Set $k_{\max}$, which denotes the number of iterations for which the algorithm is run, to a sufficiently large number. Start system simulation at the starting state, i_*, and set $s = 1$.
Loop until $k = k_{\max}$:
Step 1. Let the current state be denoted by i and the current stage be denoted by s. Select action a with a probability of $1/|\mathcal{A}(i, s)|$.
Step 2. Simulate action a. Let the next state be j in stage $(s+1)$. Let $r(i, s, a, j, s+1)$ be the immediate reward earned in the transition under the influence of action a.
Step 3. Update $Q(i, s, a)$ as follows:

$$Q^{k+1}(i,s,a) \leftarrow (1-\alpha^k)Q^k(i,s,a) + \alpha^k \times$$

$$\left[w_\tau(i,s,a,j,s+1) + \max_{b\in\mathcal{A}(j,s+1)} Q^k(j,s+1,b)\right], \quad (14)$$

Step 4. Increase k by 1. Set $s \leftarrow s+1$. If $s = N+1$, set $i = i_*$, $s = 1$, and return to Step 2. Else if $s \neq N+1$, set $i \leftarrow j$ and return to Step 2.
Termination For each $i \in \mathcal{S}$ and $s = 1, 2, \ldots, N$, select $d(i, s) \in \arg\max_{b\in\mathcal{A}(i,s)} Q(i, s, b)$. The policy (solution) generated by the algorithm is d. Stop.

We must note that the algorithm will loop between Steps 1 and 4 and will exit when $k = k_{\max}$ regardless of the value of s. Further, like in the case of the DP algorithm, as per Assumption 6, we must enforce the terminal state in stage $(N+1)$ to be reward-free. This is ensured by the fact that $Q(i, N+1, a)$ is never updated for any (i, a) pair and thus always remains at zero. Once again, the convergence of the algorithm follows in a straightforward manner from the same for Q-Learning for the SSP, and is hence relegated to the Appendix A4.

4 Numerical Results

In this section, we illustrate the use of our algorithms on instances of downside-risk-adjusted problems. We present results on three problem classes: First, we describe the use of the policy iteration algorithm on a preventive maintenance MDP where the transition probabilities are available. Thereafter, we present an analysis of the RL algorithm on an SMDP where the transition probabilities are not easy to estimate. We conclude with a numerical analysis with a finite horizon MDP via an RL algorithm. We used the MATLAB software in all our experiments, and we note that no experiment took more than 5 seconds on an Intel Pentium Processor with a speed of 2.66 GHz on a 64-bit operating system.

4.1 Infinite Horizon MDP

We consider a problem of preventive maintenance from [29]. A failure-prone production line is considered. When the line fails, it is has to be repaired, which takes the whole day. At times, the line is maintained in a preventive manner, which takes the entire day. The model developed is similar to that in [12] for which data was obtained from a firm in New York. The underlying problem here is one of determining which action to perform at the start of each day: a production or a maintenance. As the line ages, its probability of failure increases. A preventive maintenance costs less than a repair, which must be performed in case of failure. The underlying dynamics of this line will be modeled as a Markov chain, and the decision-making problem can be set up as a MDP which has two actions {*produce*, *maintain*}. We now present some notation.

- C_r: cost of a repair
- C_m: cost of a maintenance
- z: parameter that will define the failure probabilities

- *days*: number of days since last repair or maintenance

We will assume that when the system is repaired or maintained, it is as good as new. As stated above, we use the model in [12] for the failure dynamics, where the state of the MDP is defined by *days* and transition probability law under the production action is defined as follows: $p(days, produce, days + 1) = z^{days}$ and $p(days, produce, 0) = 1 - z^{days}$ for $days = 0, 1, \ldots, D$, where D is the maximum value for the variable *days*; all other transition probabilities for the action will equal 0. In practice, a suitable positive integer will be chosen for the value of D such that at D, $p(D, produce, 0) \approxeq 1$; hence, we will assume that $p(D, produce, 0) = 1$ and $p(D, produce, D) = 0$. For the maintenance action, the transition probability law will be as follows: for $days = 0, 1, \ldots, D$, $p(days, maintain, 0) = 1$ and other values of the transition probability will equal 0. Since, the transition to state 0 implies a failure, under action *produce*, and a maintenance, under action *maintain*, we have the following for the transition reward matrix: $r(days, produce, 0) = -C_r$ and $r(days, maintain, 0) = -C_m$ for $days = 0, 1, \ldots, D$; all other values of $r(.,.,.)$ equal 0. We now present an illustrative example:

Example A: We use the following values in this experiment: $C_m = 3$, $C_r = 10$, $z = 0.99$, $\theta = 10$, $\tau = -5$, and $D = 20$. Using the policy iteration algorithm presented above, we obtain the following solution after 6 iterations: Action *produce* is optimal in states $days = 0, 1, \ldots, 5$ and from $days = 6$ onwards, action *maintain* is optimal. We compared this to the risk-neutral situation where $\theta = 0$, for which the optimal solution is: action *produce* in states $days = 0, 1, \ldots, 7$ and from $days = 8$ onwards, action *maintain*. It is clear thus that the risk-averse solution recommends maintenance at an earlier age (6 days) in comparison to the age recommended by the risk-neutral solution (8 days). A plot of the behavior of the algorithm, which shows the value of ϕ in each iteration of the algorithm, is presented in Fig. 1.

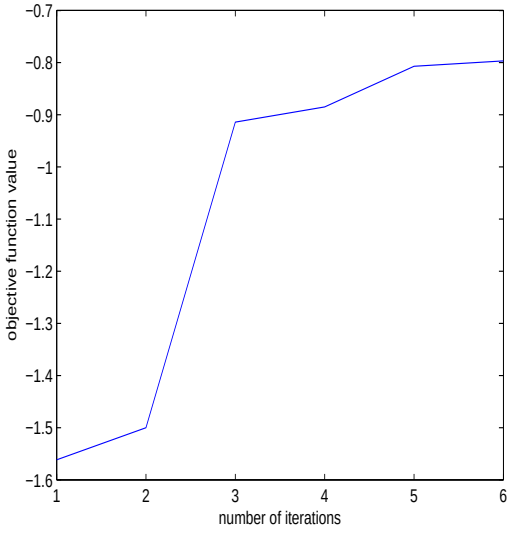

Figure 1 The progression of policy iteration: The objective function here is ϕ.

4.2 Infinite Horizon SMDP

We now consider a case where the RL algorithm will be used on an infinite horizon SMDP from [30]; the transition probabilities for problems of this nature are not easy to evaluate, and hence simulation-based algorithms (e.g., RL) form a useful alternative. A production-inventory system, i.e., a machine with a finished product buffer is considered. The buffer stores the product until the demand from the customer arrives. The demand has a size of 1, while the machine increases the size of the buffer by 1 when it produces a part successfully. There is an upper limit, S, on how much the buffer can store. The machine takes a break from production when this upper limit is reached, and it remains on vacation until the buffer falls to a pre-determined level, s. The input random variables for this model are: time for producing a part (production time), the time between failures, the time for a repair, the time between demand arrivals, and the time for a maintenance. The "age" of the machine is determined by the number of units produced since last repair or maintenance. The state-space for the SMDP will be defined as: $\{\mathsf{b}, \mathsf{c}\}$, where c denotes the number of parts produced since last repair or maintenance, and b stands for the number of parts currently inside the buffer. There are two actions that the decision maker can select from: {*Produce, Maintain*}. The action is to be selected at the end of a production cycle, i.e., when one unit is produced.

Example B: We use the following data-set from the literature [30]: $(\mathsf{S}, \mathsf{s}) = (3, 2)$. Further, we will use the following notation: $Expo(\lambda)$ will denote an exponential distribution with a mean of $1/\lambda$, while $Erl(shape, scale)$ will denote an Erlang distribution with the shape and scale parameters as specified within brackets and $Unif(a, b)$ will denote the uniform distribution with a as the minimum and b as the maximum. In our experiment, we use: $Expo(1/10)$ for time between arrivals, $Erl(8, 12.5)$ for the time between failures, $Erl(2, 100)$ for the time for repair, $Erl(8, 1.25)$ for the production time, and $Unif(5, 20)$ for the maintenance time. $C_r = \$5$, $C_m = \$2$, and profit per sale of one unit is \$1. The policy turns out to have a threshold nature, i.e., for $i = 1, 2, 3$, when the buffer equals i, the production action is chosen as long as the production count c is less than c_i and maintain action when $\mathsf{c} \geq \mathsf{c}_i$. We also use $\tau = -\$3$, a value less than the cost of maintenance, and $\theta = 10$.

When the RL algorithm is run for the downside-risk-adjusted objective, it returns $c_1 = 3$, $c_2 = 4$, and $c_3 = 7$; when buffer is empty, the action is to produce regardless of the size of the buffer. The value of the average reward, ρ, equals 0.0263 while the probability of failure, which is essentially the downside risk here, equals 0.0041. For the risk-neutral case, which is obtained by setting $\theta = 0$, the thresholds are $c_1 = 5$, $c_2 = 5$, and $c_3 = 6$; when the buffer is empty, the action is to produce regardless of the size of the buffer. Also, $\rho = 0.0342$ and the downside risk is 0.0106 for the risk-neutral case. As is clear, when the algorithm accounts for the downside risk, the latter gets lowered but the average reward also falls, which is as expected. In all the

experiments, we used $\eta = 0.99$ and the following rules for the step-sizes: $\alpha^k = 150/(300+k)$ and $\beta^k = 10/(300+k)$. For small problems, it is possible to run the algorithm with different values for η, but for larger problems, one must guesstimate a value.

4.3 Finite Horizon MDP

The goal here is to numerically analyze a simple two-stage finite horizon MDP for which an optimal policy can be identified via inspection. The following example will also be used as a test case for the RL algorithm and to explain the usefulness of the downside risk criterion.

Example C: The problem has 2 stages, i.e., $N = 2$ and the initial state is unique. Fig. 2 provides the data for this example. We will have eight policies, shown in Table I. Table II shows the expected total reward for each policy and the total downside risk, as well as the risk-adjusted objective function $\phi(1,1)$. We provide details of the values in the second and third rows of Table II — in order to explain how these calculations are performed. When policy 1 is chosen, it is clear from the data in Fig. 2 that the total expected reward (TER) for this policy will be:

$$TER = 0.7(10+4) + 0.3(2+5) = 11.9.$$

We use $\tau = 6$ and $\theta = 10$ in our experiments. Then, the downside risk (DR) of any policy will be

$$DR = Prob(X_1 < \tau) + Prob(X_2 < \tau),$$

where X_s denotes the immediate reward earned in stage s for $s = 1, 2$. Then, the downside risk for policy 1 will be:

$$DR = 0.3 + 1 = 1.3.$$

As a result, the objective function for this policy will be: $\phi(1,1) = 11.9 - \theta(1.3) = 11.9 - 10(1.3) = -1.1$. Next, we consider policy 2. Here, the corresponding values will be:

$$TER = 0.5(6+4) + 0.5(7+5) = 11;$$

$$DR = 0 + 1 = 1; \text{ and}$$

$$\phi(1,1) = 11 - 10(1) = 1.$$

From Table II, by evaluating all the policies, it is clear that policies 4 and 8 are optimal, i.e., choose action 2 in stage 1, action 2 in state 1 of stage 2, and either action in state 2 of stage 2. For the risk-neutral case, the optimal solutions, which maximize the expected total reward, are policies 3 and 6.

We ran the RL algorithm on a simulator of this finite horizon MDP, and obtained the optimal policy in 1000 iterations with the following Q-values: $Q(1,1,1) = 6.0699$, $Q(1,1,2) = 7.5171$, $Q(1,2,1) = 0.8796$, $Q(1,2,2) = 1.8328$, $Q(2,2,1) = 1.1614$, and $Q(2,2,2) = 1.2941$. In other words, policy 4, which is one of the optimal policies, is returned. For the risk-neutral case, policies 3 and 6 are optimal. When the algorithm is run for the risk-neutral case, policy 3 is returned.

In the experiment above, we further use target semi-variance as a risk metric [23], in order to examine the resulting policy. The risk-adjusted reward using target semi-variance as a risk metric can be defined as:

$$w(i,s,a,j,s+1) = r(i,s,a,j,s+1) - \theta \times$$

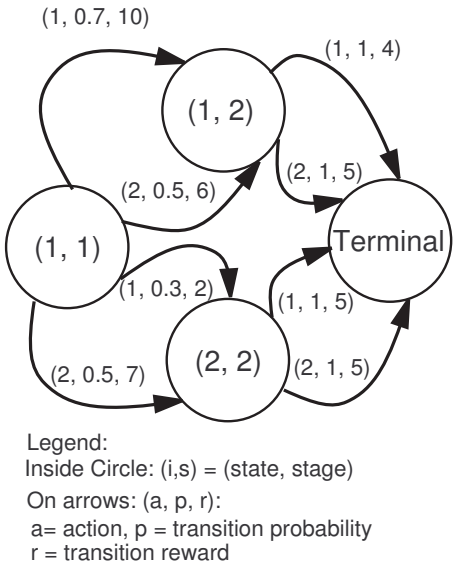

Figure 2 The figure shows the transition probabilities and rewards for the different actions in Example C in which $\tau = 6$ and $\theta = 10$.

$$\left([\tau - r(i,s,a,j,s+1)]^+\right)^2, \tag{15}$$

where $[a]^+ \equiv \max(0,a)$. Note that the above is the counterpart of target semi-variance for Equation (10); thus, the target semi-variance computes the square of the deviation of the immediate reward *below* the target, τ, i.e., the target semi-variance is the square of the semi-deviation below the target. Note that when the immediate reward exceeds the target, the deviation equals 0. Using the definition above for $w(.,.,.,.)$, one can then define $\bar{w}(.,.)$ via Equation (11) and the objective function for the target semi-variance in a finite horizon MDP via Definition 3. The finite horizon RL algorithm for the downside risk case with the new definition for $w(.,.,.,.)$ will now serve as the target semi-variance counterpart.

We used $\theta = 1$ in our experiment for target semi-variance and ran the algorithm for 1000 iterations in a simulator for the finite horizon MDP. The following Q-values were obtained, which yielded the same policy as obtained in the downside risk case: $Q(1,1,1) = 19.7848$, $Q(1,1,2) = 21.7516$, $Q(1,2,1) = 17.0010$, $Q(1,2,2) = 20.6861$, $Q(2,2,1) = 16.4804$, and $Q(2,2,2) = 17.2725$.

The reason for using the value of 1 for θ in the semi-variance case, in contrast to the value of 10 used in the downside risk case, was that downside risk is a probability, and as such, its value is upper bounded by 1; however, target semi-variance can acquire large values (because of the squaring involved), and therefore a large value for θ can excessively amplify the importance of the risk. Another way of explaining this would be to say that using an excessively large value for the penalty factor, θ, can cause the problem to ignore the average reward and only emphasize the risk, which can lead to a solution whose risk is low, but whose average reward is also low; this is unattractive from all aspects. Hence, the penalty factor must be *tuned* in order to obtain a sensible policy in practice. Our experiments

suggested that for the problem we considered, using values such as 10 for θ for the target semi-variance risk produced unattractive policies; the appropriate values for target semi-variance are in the range of $(1, 3)$, whereas for the downside risk case, θ must lie in between 7 and 12. Thus, our experiments suggested that θ must be tuned to obtain useful solutions.

Table I Definitions of the eight policies; the numbers provided in the second through the fourth column denote actions for the stage-state combination

Policy	Stage 1	Stage 2	Stage 2
	State 1	State 1	State 2
1	1	1	1
2	2	1	1
3	1	2	2
4	2	2	2
5	1	1	2
6	1	2	1
7	2	1	2
8	2	2	1

Table II The total expected reward (TER), the downside risk (DR), and the objective function ($\phi(1, 1)$)

Policy	TER	DR	$\phi(1,1)$
1	11.9	1.3	−1.1
2	11	1	1
3	12.6	1.3	−0.4
4	11.5	1	1.5
5	11.9	1.3	−1.1
6	12.6	1.3	−0.4
7	11	1	1
8	11.5	1	1.5

5 Conclusions

The intersection of risk and MDPs has been studied for many years now. There is a significant body of literature on the exponential utility function and MDPs. Further, papers written on the risks of revenues falling below prespecified targets in MDPs have mostly addressed the issue of *minimizing* the so-called threshold probability or downside risk, which is at one extreme of the solution spectrum. For instance in the maintenance problem, minimizing the risk would lead to an impractical policy in which maintenance would be recommended on the first day, thus not allowing any production. In this paper, our goal was much more pragmatic — to develop solution techniques that addressed the issue of *reducing* the downside risk and at the same time obtaining a reasonably high value for the average reward. To this end, we used a downside-risk-adjusted objective function and developed the underlying Bellman equations for SMDPs/MDPs on the infinite horizon, as well as MDPs on the finite horizon under some assumptions. Thereafter, we developed dynamic programming algorithms and reinforcement learning algorithms. We concluded with a numerical analysis to demonstrate the usefulness of the downside risk penalty. Our theoretical and numerical analysis provides encouraging results.

A number of directions for future work can be envisioned. First, the numerical tests should be extended to problems of a larger scale. Second, the problem can be analyzed after the key assumptions made in this analysis, on the nature of the underlying Markov chains for the infinite horizon and the nature of policies and uniqueness of starting states for the finite horizon, are relaxed.

Acknowledgement

The authors would like to acknowledge the two anonymous reviewers and the Associate Editor for finding critical typos in the paper and making suggestions for improvement.

References

[1] R. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, Cambridge, MA, USA, 1998.

[2] D.P. Bertsekas and J. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Belmont, MA, USA, 1996.

[3] P. Balakrishna, R. Ganesan, and L. Sherry. Accuracy of reinforcement learning algorithms for predicting aircraft taxi-out times: A case-study of Tampa bay departures. *Transportation Research Part C: Emerging Technologies*, 18(6):950–962, 2010.

[4] Z. Sui, A. Gosavi, and L. Lin. A reinforcement learning approach for inventory replenishment in vendor-managed inventory systems with consignment inventory. *Engineering Management Journal*, 22(4):44–53, 2010.

[5] P. Abbeel, A. Coates, T. Hunter, and A.Y. Ng. Autonomous autorotation of an RC helicopter,. In *International Symposium on Robotics*, 2008.

[6] R. Howard and J. Matheson. Risk-sensitive Markov decision processes. *Management Science*, 18(7):356–369, 1972.

[7] M. Rabin. Risk aversion and expected utility theory: A calibration theorem. *Econometrica*, 68:1281–1292, 2000.

[8] P. Whittle. *Risk-Sensitive Optimal Control*. John Wiley, NY, USA, 1990.

[9] J. Filar, L. Kallenberg, and H. Lee. Variance-penalized Markov decision processes. *Mathematics of Operations Research*, 14(1):147–161, 1989.

[10] M. Sobel. The variance of discounted Markov decision processes. *Journal of Applied Probability*, 19:794–802, 1982.

[11] M. Sato and S. Kobayashi. Average-reward reinforcement learning for variance penalized Markov decision problems. In *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*, pages 473–480, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[12] A. Gosavi. A risk-sensitive approach to total productive maintenance. *Automatica*, 42:1321–1330, 2006.

[13] A. Gosavi. Variance-penalized Markov decision processes: Dynamic programming and reinforcement learning techniques. *International Journal of General Systems*, 43(6):649–669, 2014.

[14] A. Gosavi. Reinforcement learning for model building and variance-penalized control. In *2009 Winter Simulation Conference (WSC'09)*, Austin (TX), USA, 2009.

[15] O. Mihatsch and R. Neuneier. Risk-sensitive reinforcement learning. *Machine Learning*, 49(2-3):267–290, 2002.

[16] P. Geibel. Reinforcement learning via bounded risk. In *ICML01*, pages 162–169. Morgan Kaufman, 2001.

[17] M. Heger. Considerations of risk in reinforcement learning. In *Proceedings of 11th International Conference on Machine Learning*, pages 105–111. Morgan Kaufmann, 1994.

[18] Y. Chen and J. Jin. Cost-variability-sensitive preventive maintenance considering management risk. *IIE Transactions*, 35(12):1091–1102, 2003.

[19] C. Barz and K. Waldmann. Risk-sensitive capacity control in revenue management. *Math. Meth. Oper. Res.*, 65:565–579, 2007.

[20] K. Chung and M. Sobel. Discounted MDPs: Distribution functions and exponential utlity maximization. *SIAM Journal of Control and Optimization*, 25:49–62, 1987.

[21] M. Bouakiz and Y. Kebir. Target-level criterion in Markov decision processes. *Journal of Optimization Theory and Applications*, 86:1–15, 1995.

[22] C. Wu and Y. Lin. Minimizing risk models in Markov decision processes with policies depending on target values. *Journal of Mathematical Analysis and Applications*, 231:47–67, 1999.

[23] A. Gosavi. Target-sensitive control of Markov and semi-Markov processes. *International Journal of Control, Automation, and Systems*, 9(5):1–11, 2011.

[24] A. Gosavi, S.K. Das, and S.L. Murray. Beyond exponential utility functions: A variance-adjusted approach for risk-averse reinforcement learning. In *Adaptive Dynamic Programming and Reinforcement Learning (ADPRL), 2014 IEEE Symposium on*, pages 1–8, 2014.

[25] D.P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena, Belmont, 1995.

[26] M. L. Puterman. *Markov Decision Processes*. Wiley Interscience, New York, 1994.

[27] T.K. Das, A. Gosavi, S. Mahadevan, and N. Marchalleck. Solving semi-Markov decision problems using average reward reinforcement learning. *Management Science*, 45(4):560–574, 1999.

[28] J. Baxter and P. Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence*, 15:319–350, 2001.

[29] A. Parulekar. A downside risk criterion for preventive maintenance. Master's thesis, University at Buffalo, The State University of New York, 2006.

[30] T.K. Das and S. Sarkar. Optimal preventive maintenance in a production inventory system. *IIE Transactions*, 31:537–551, 1999.

Appendix

A1: Relative Value Iteration: We first present a relative value iteration algorithm for MDPs in which any state in the system is first chosen as the distinguished state.
Step 1: Select any arbitrary state from $\mathcal{S}$ to be a distinguished state i^*. Set $k = 1$, and select arbitrary values for the vector $\vec{J}^1$. Specify a small, but positive termination value for ϵ.
Step 2: Compute for each $i \in \mathcal{S}$: $J^{k+1}(i) =$

$$\max_{a \in \mathcal{A}(i)} \left[\sum_{j \in \mathcal{S}} p(i,a,j)[w_\tau(i,a,j) + J^k(j)] \right].$$

When calculations are complete for every state above, set $\phi^k = J^{k+1}(i^*)$.
Step 3: For each $i \in \mathcal{S}$, calculate:

$$J^{k+1}(i) \leftarrow J^{k+1}(i) - \phi^k.$$

Step 4: If

$$||\vec{J}^{k+1} - \vec{J}^k||_\infty < \epsilon,$$

go to Step 5. Otherwise, increment k by 1 and return to Step 2.
Step 5: For each $i \in \mathcal{S}$, choose $d(i) \in$

$$\arg\max_{a \in \mathcal{A}(i)} \left[\sum_{j \in \mathcal{S}} p(i,a,j)[w_\tau(i,a,j) + J^k(j)] \right].$$

The ϵ-optimal policy is d; ϕ^k is the optimal objective function's estimate.

A2: Convergence of the RL Algorithm for infinite horizon MDPs

Theorem 4 *Assume that the iterates α^k and β^k satisfy the following condition:* $\lim_{k\to\infty} \frac{\beta^k}{\alpha^k} = 0$. *Then, under Assumptions 1–2 and if the step sizes, α^k and β^k satisfy 3, the sequence of iterates in the RL algorithm for infinite horizon,*

$\{Q^k(.,.)\}_{k=1}^{\infty}$, converges with probability 1 to the unique solution of Equation (9) and leads to an optimal solution of the SMDP, as $k \to \infty$.

Proof The proof follows directly from Theorem 7 in [23] by noting that the function $w(.,.,.)$ in [23] would have to be replaced by the function $w(.,.,.)$ defined here. ∎

A3: Convergence of the DP Algorithm for finite horizon MDPs

Theorem 5 *Under Assumptions 4–6, the sequences of iterates, $\{J^k(.)\}_{k=1}^{\infty}$, generated by the finite horizon DP algorithm converges to the unique solution of the Bellman optimality equation, i.e., Equation (12), as $k \to \infty$.*

Proof We only sketch the proof, because it follows directly from that of the convergence of the SSP. Essentially, the finite horizon MDP can be treated as a special case of the SSP under Assumptions 4–6. This can be done by treating the state-stage pair as the state in the SSP. Proposition 2.1 (b) of [2] shows the convergence of the value iteration algorithm for the SSP; when $r(.,.,.)$ in that result is replaced by $w(.,.,.)$ defined in this paper, the result is immediate. ∎

A4: Convergence of the RL Algorithm for finite horizon MDPs

Theorem 6 *Under Assumptions 3-6, the sequence of iterates in the RL algorithm for finite horizon, $\{Q^k(.,.)\}_{k=1}^{\infty}$ converges with probability 1 to the unique solution of Equation (13).*

Proof Under Assumptions 3–6, the convergence is immediate from Prop. 5.5 (a) in [2] after noting that the finite horizon problem can be treated as a special case of the SSP and replacing the function $r(.,.,.)$ by $w(.,.,.)$. ∎

Abhijit Gosavi received his B.E. in Mechanical Engineering from the Jadavpur University, India, in 1992 and his M.Tech in Mechanical Engineering from the Indian Institute of Technology, Madras, India in 1995. He received a Ph.D. in Industrial Engineering from University of South Florida in 1999. Currently, he is an Associate Professor in the Department of Engineering Management and Systems Engineering at Missouri University of Science and Technology.

He has published more than 60 refereed journal and conference papers. His research interests cover simulation-based optimization, Markov decision processes, productive maintenance, and revenue management.

Dr. Gosavi has received research funding awards from the National Science Foundation of the United Stated of America and numerous other agencies. He is a member of IIE, ASEM, and INFORMS.

Email: gosavia@mst.edu

Anish Parulekar obtained an M.S. in Industrial Engineering from the Department of Industrial and Systems Engineering at the University of Buffalo, State University of New York in 2006 and a B.E. in Mechanical Engineering from University of Mumbai, India in 2004. He currently serves as a Deputy Vice President and the Head of Marketing Analytics in Axis Bank, Mumbai, India.

His research interests are in risk, computing, and Markov control.

Email: anish.parulekar@axisbank.com