Multi-Objective Approaches to Markov Decision Processes with Uncertain Transition Parameters

Dimitri Scheftelowitsch
Informatik IV, TU Dortmund
Dortmund, Germany
dimitri.scheftelowitsch@cs.tu-dortmund.de

Peter Buchholz
Informatik IV, TU Dortmund
Dortmund, Germany
peter.buchholz@cs.tu-dortmund.de

Vahid Hashemi
Saarland University – Computer Science
Saarbrücken, Germany
hashemi@depend.uni-saarland.de

Holger Hermanns
Saarland University – Computer Science
Saarbrücken, Germany
hermanns@depend.uni-saarland.de

Abstract

Markov decision processes (MDPs) are a popular model for performance analysis and optimization of stochastic systems. The parameters of stochastic behavior of MDPs are estimates from empirical observations of a system; their values are not known precisely. Different types of MDPs with uncertain, imprecise or bounded transition rates or probabilities and rewards exist in the literature.

Commonly, analysis of models with uncertainties amounts to searching for the most robust policy which means that the goal is to generate a policy with the greatest lower bound on performance (or, symmetrically, the lowest upper bound on costs). However, hedging against an unlikely worst case may lead to losses in other situations. In general, one is interested in policies that *behave well* in *all* situations which results in a multi-objective view on decision making.

In this paper, we consider policies for the expected discounted reward measure of MDPs with uncertain parameters. In particular, the approach is defined for bounded-parameter MDPs (BMDPs) [8]. In this setting the worst, best and average case performances of a policy are analyzed simultaneously, which yields a multi-scenario multi-objective optimization problem. The paper presents and evaluates approaches to compute the *pure* Pareto optimal policies in the value vector space.

CCS Concepts

• **Theory of computation → Theory and algorithms for application domains**; • **Applied computing → Operations research**;

ACM Reference Format:
Dimitri Scheftelowitsch, Peter Buchholz, Vahid Hashemi, and Holger Hermanns. 2017. Multi-Objective Approaches to Markov Decision Processes with Uncertain Transition Parameters. In *VALUETOOLS 2017: 11th EAI International Conference on Performance Evaluation Methodologies and Tools, December 5–7, 2017, Venice, Italy.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3150928.3150945

VALUETOOLS 2017, December 5–7, 2017, Venice, Italy

ACM ISBN 978-1-4503-6346-4/17/12…$15.00
https://doi.org/10.1145/3150928.3150945

1 Introduction

Markov decision processes (MDPs) are a common tool to describe decision situations in many different contexts such as performance optimization and planning [2, 3, 13, 16]. The general idea is to specify a system by means of different *states* in which it can be, *actions* which a decision maker can perform to affect the (probabilistic) future behavior, and *rewards* or *costs* that depend on the state and decision such as energy costs of a server starting up or the amount of users served in a queue if a service is active. After an action has been chosen, the system changes its state depending on the action and the current state but not on the past behavior; transitions are, in general, randomized and defined by the system's properties.

However, modeling a physical or an artificial system suffers from several limitations. Most prominent is the inherent loss of precision that is introduced by measurement errors and discretization artifacts which necessarily happen due to incomplete knowledge about the system behavior. So the true probability distribution to be associated with transitions is in most cases uncertain and instead can be given by either external parameters or confidence intervals. To account for the latter, the MDP model has been extended to bounded-parameter MDPs (BMDPs) [8] or the slightly more general classes of MDPs with incomplete or uncertain transition probabilities [19, 25]. In these classes of MDPs, best-case and worst-case policies for expected discounted reward measures have been considered. These policies result in upper and lower bounds for the value vector that contains the discounted accumulated reward gained after starting in the different states of the process.

Robust optimization techniques [29] are useful to derive policies that hedge against model uncertainties. In particular, these *robust* policies optimize the expected discounted reward against the worst possible realization of the uncertainty. From the viewpoint of a potential decision maker, however, a robust solution may be overly pessimistic. A decision maker is often interested in a solution that might be not fully robust, but instead could have an acceptable worst-case behavior while retaining good best-case and

average-case (for a properly defined concept of "average") performance. This property may be formalized in several alternative ways; most notably, there are several competing definitions of "almost robust" solutions [12]. A further optimization goal in this context may be the probability of reaching a certain performance bound, assuming some probability distribution over the uncertainty set. Unfortunately, the latter seems to be inherently hard, especially for uncertain MDPs [20].

Thus a promising approach considers not only one extremal measure (either the upper or the lower bound), but at least both bounds, and possibly the expectation, and optimizes all of them simultaneously. This means doing *multi-objective optimization* which yields several mutually incomparable, so-called *non-dominated* policies for the uncertain MDP from which the user may select the one which has the most suitable performance measures. This may be in particular of interest if the differences between the optimal policies in the different scenarios are sufficiently large to consider different tradeoffs besides the obvious extremal points. In this paper, we develop methods to compute all pure non-dominated policies for a given uncertain MDP in a specific uncertainty setting which can be generalized to other notions of parameter uncertainty. This problem which is basically an instance of *multi-scenario optimization* asks to compute policies which are optimal in the presence of trade-offs between several conflicting objectives.

It is worthwhile to mention that for the considered multi-objective optimization problem, the number of optimal, i.e., mutually incomparable and non-dominated solutions may be exponentially high, which is a structural feature of the problem, and computing all of these solutions might take a prohibitive amount of time. However, for "stationary" decision-making problems which only rarely vary with time, we argue that the amount of time invested in finding an optimal policy is often negligible when compared to the actual performance of the policy implementation.

Related work. The Bounded-parameter MDP (BMDP) model is introduced and widely discussed in [8]. BMDPs are a specific subclass of MDPs with uncertain or imprecise transition probabilities proposed by [19] and [25]. The methods extend to more general notions of uncertainty in MDPs such as convex uncertainty sets discussed by [15]; further aspects of parameter uncertainty in MDPs are covered in [6, 11, 29]. However, in almost all cases, the goal is to compute a robust policy which ensures the best possible behavior in the worst case. In many scenarios however, we may have a multidimensional reward function and hence search for a policy which simultaneously maximizes all reward dimensions. To account for the latter, multi-objective Markov decision processes are discussed in [1, 5, 14, 26, 28]. The extension to multi-objective Markov decision processes under bounded parameter uncertainty has recently been investigated in [9]. In this paper, we target another facet of multi-objective multi-scenario optimization for MDPs with parameter uncertainty where the goal is to simultaneously optimize the worst, best and average case behavior.

There exist numerous applications of MDPs and uncertainty modeling in the performance optimization world. They include decision support in medical screening procedures [3] and product line design [2]. Applications and the general formulation of multi-scenario optimization are given in [27].

Our Contribution. This work considers the multi-scenario multi-objective optimization problem for BMDPs where the multiple scenarios correspond to the maximal, minimal, and average performance measures of a given policy in the uncertainty set of BMDPs. The notion of average performance is given by slightly extending the existing BMDP formalism and defining a designated average MDP in the uncertainty set. In particular, we define the average MDP by introducing a probability distribution over the uncertainty set and deriving the expected value. This has the advantage of possible further applications such as percentile optimization.

In summary, the main contributions of this paper are threefold. First, we provide an exact algorithm that computes the desired set of pure Pareto optimal policies, albeit at a (almost certainly) prohibitively high computational cost. Second, we design a heuristic that is efficient in the sense that it computes a set of mutually non-dominated policies that are likely to be Pareto optimal with reasonable time complexity. Finally, we develop a prototype tool and apply it to some case studies to show the effectiveness of our approach.

2 Mathematical Preliminaries

We start with common notations and formalisms used in this paper.

For a matrix M, we denote by $m_{i,j}$ the entry in row i and column j. Vector identifiers, such as $\mathbf{v}$, appear in bold script, to distinguish them from scalars and matrices. For a natural number n, we designate by $[n]$ the set $\{1, 2, \ldots, n\}$. For multi-dimensional identifiers, such as matrices or vectors, the order relations $\leq$ or $\geq$ are performed componentwise. Additionally, we define $\mathbf{v} >_P \mathbf{v}'$ to be short for $\mathbf{v} \geq \mathbf{v}' \wedge \mathbf{v} \neq \mathbf{v}'$.

We next motivate and formally introduce the modelling formalism considered in this paper.

In a Markov decision process, transition probabilities and rewards are estimates resulting from measurements or expert opinion. This implies that there is always uncertainty about the parameters of the model and also about the behavior of the real system according to some policy that has been derived from the MDP. The class of stochastic bounded-parameter MDPs (SBMDPs) includes this uncertainty by considering intervals rather than point estimates for the parameters of MDPs and defining a probability distribution over the uncertainty set. We shall use the probability distribution in order to derive the "average" MDP and then work with this MDP; for the sake of generality, we would like to define the formalism here completely with the probability distribution.

*Definition 2.1 (**Stochastic bounded-parameter Markov process**).* A *stochastic bounded-parameter Markov decision process* (SBMDP) is a tuple $(S, A, T_{\updownarrow}, R_{\updownarrow}, Pr)$ where $S = [n]$ is a (finite) state space, $A = [m]$ is a (finite) set of *actions*, $T_{\updownarrow} = ((P^1_{\downarrow}, P^1_{\uparrow}), \ldots, (P^m_{\downarrow}, P^m_{\uparrow}))$ is a set of m matrix pairs where for each $a \in A$, $0 \leq P^a_{\downarrow} \leq P^a_{\uparrow}$, $P^a_{\downarrow}\mathbb{1} \leq \mathbb{1} \leq P^a_{\uparrow}\mathbb{1}$, $R_{\updownarrow} = ((\mathbf{r}^1_{\downarrow}, \mathbf{r}^1_{\uparrow}), \ldots, (\mathbf{r}^m_{\downarrow}, \mathbf{r}^m_{\uparrow}))$ is a set of m reward vector pairs where for each $a \in A$, $0 \leq \mathbf{r}^a_{\downarrow} \leq \mathbf{r}^a_{\uparrow}$ and $Pr = \left\{(p_{a,r} : \mathbb{R}^n \to \mathbb{R}, p_{a,P} : R^{n,n} \to \mathbb{R}) \mid a \in A\right\}$ is a probability measure on the rewards and the transition matrices for each action.

We denote by $\mathbf{p}^a_{\updownarrow s}$ all vectors $\mathbf{p}^a_s$ such that $p^a_{\downarrow s,s'} \leq p^a_{s,s'} \leq p^a_{\uparrow s,s'}$ for all $s' \in [n]$ and $\sum_{s'=1}^n p^a_{s,s'} = 1$. Similarly $\mathbf{r}^a \in \mathbf{r}^a_\updownarrow$ specifies all vectors $\mathbf{r}^a$ such that $r^a_{\downarrow s} \leq r^a_s \leq r^a_{\uparrow s}$.

It is worthwhile to note that the model of SBMDPs extends the formalism of BMDPs introduced by [8] with a probability measure on the possible transition matrices and reward vectors, so as to enable to take the "average performance" into consideration by deriving expected values for the transition matrices and rewards. This is different from the expected value of the value vector under the probability distribution, as here the expectation operator is applied on the model and serves to define an additional optimization objective, in addition to the upper and lower bounds for the value vectors.

To optimize the performance of a (SB)MDP, a *decision rule* or *policy* is needed. Formally, a policy is a function $f\colon S^{\mathbb{N}} \to \Delta(A)$ where $S^{\mathbb{N}}$ is the set of (finite) *histories* of states and $\Delta(A)$ is the set of probability distributions on A. We call a policy f *pure* if it is stationary, i.e., it depends only on the current state, and deterministic, i.e, it always maps a history to a Dirac distribution, i.e., $f(\cdot, a) \in \{0,1\}$. We use f for general policies and π for stationary and pure policies. We denote by $\pi(s) \in A$ the action that is chosen in state s using policy π, if π is pure.

Moreover, a stationary policy π induces a Markov reward process with transition matrix $P^{(\pi)}$ and reward vector $\mathbf{r}^{(\pi)}$. For a transition matrix P and state s, we denote by $\mathbf{p}_s$ the s-th row of P. Accordingly, we denote by $\mathbf{p}^{\pi(s)}_s$ the s-th row of $P^{(\pi)}$ and by $r^{\pi(s)}_s$ the reward of state s under policy π. For pure policies, we define a Hamming distance measure d by $d(\pi,\pi') = |\{s \in S \mid \pi(s) \neq \pi'(s)\}|$.

The *expected discounted reward* for an infinite horizon is taken as performance metric. For a MDP, a given policy f defines a sequence of transition matrices $(P^{(f)}_t)_{t\in\mathbb{N}}$ and reward vectors $(\mathbf{r}^{(f)}_t)_{t\in\mathbb{N}}$ where $P^{(f)}_t$ resp. $\mathbf{r}^{(f)}_t$ is the transition matrix resp. the reward in the t-th time step. Using this sequence, we derive a probability distribution on sequences of states and corresponding rewards, from which the expected discounted reward is defined by $\mathrm{Ex}\sum_{t\in\mathbb{N}} \gamma^{t-1}\mathbf{r}^{(f)}_t$.

A *value vector* $\mathbf{v}$ collects the computed expected discounted reward for all states where each entry equals the expected discounted reward one obtains starting from the respective state. The mathematical properties of expected discounted rewards on which we shall rely in the sequel are widely discussed by [16].

We assume that the goal is the maximization of the expected discounted reward. For a SBMDP policies assuring best and worst case behavior can be determined by solving the following Bellman equations for the *value vectors* $\mathbf{v}_\downarrow, \mathbf{v}_\uparrow$.

$$\begin{aligned} v_{\downarrow s} &= \max_{a\in A}\left(r^a_{\downarrow s} + \gamma \min_{\mathbf{p}^a_s \in \mathbf{p}^a_{\updownarrow s}} \left(\mathbf{p}^a_s \mathbf{v}_\downarrow\right)\right) \\ v_{\uparrow s} &= \max_{a\in A}\left(r^a_{\uparrow s} + \gamma \max_{\mathbf{p}^a_s \in \mathbf{p}^a_{\updownarrow s}} \left(\mathbf{p}^a_s \mathbf{v}_\uparrow\right)\right) \end{aligned} \tag{1}$$

As shown by [8], the optimal policies are pure. We denote them by $\pi_\downarrow$ and $\pi_\uparrow$, respectively.

Additionally, we are interested in the "average-case" properties of SBMDPs and a policy that maximizes the performance under the "average-case" assumption. The reasoning here is fairly straightforward: Since the probability distribution for the transition matrices is known, we can define for each action $a \in A$, $\overline{P}^a = \mathrm{Ex}\,[P^a] = \int_{P^a_\updownarrow} P \cdot p_{a,P}(P)\,\mathrm{d}P$ and $\overline{\mathbf{r}}^a = \mathrm{Ex}[\mathbf{r}^a] = \int_{\mathbf{r}^a_\updownarrow} \mathbf{r}\cdot p_{a,r}(\mathbf{r})\,\mathrm{d}\mathbf{r}$, where p_a are probability densities on transition matrices and reward vectors for action a. The Bellman equations become

$$\overline{v}_s = \max_{a\in A}\left(\overline{r}^a_s + \gamma \overline{\mathbf{p}}^a_s \overline{\mathbf{v}}\right) \tag{2}$$

which is a standard MDP problem. The optimal policy is pure and is denoted as $\overline{\pi}$. For a policy f, by $\mathbf{v}^{(f)}_\downarrow$, $\mathbf{v}^{(f)}_\uparrow$ and $\overline{\mathbf{v}}^{(f)}$ we designate the lower, upper and average value vectors, furthermore, the triple $(\mathbf{v}^{(f)}_\downarrow, \overline{\mathbf{v}}^{(f)}, \mathbf{v}^{(f)}_\uparrow)$ is designated by $\mathbf{v}^{(f)}$, if not mentioned otherwise. Obviously $\mathbf{v}^{(f)}_\downarrow \leq \overline{\mathbf{v}}^{(f)} \leq \mathbf{v}^{(f)}_\uparrow$ and also $\mathbf{v}_\downarrow \leq \overline{\mathbf{v}} \leq \mathbf{v}_\uparrow$ hold, as $\mathbf{v}^{(f)}_\downarrow$ and $\mathbf{v}^{(f)}_\uparrow$ are the minimal resp. maximal value vectors for f over all MDPs in the uncertainty set. Usually $\pi_\downarrow$, $\pi_\uparrow$, and $\overline{\pi}$ differ and a compromise between two or all three goals has to be found.

Before we discuss specifics, we would like to mention a possible alternative approach in this setting. It is possible to consider the SBMDP formalism in a Bayesian setting, where the information from the probability distribution over the uncertainty set of MDPs is taken into account completely, thus seeking for $\max_f \mathrm{Ex}_M \mathbf{v}^{(f)}_M$ where $\mathbf{v}^{(f)}_M$ is the value vector of the policy f in the MDP M. This approach can be undertaken in the case where it is possible to integrate over the probability distribution over $M_\updownarrow$ easily.

Here, we explicitly choose the multi-objective approach mainly for the reason of larger applicability. Even if a Bayesian approach can be motivated from the standpoint of stochastic BMDPs, one can easily see that any reasonably good solution in the multi-objective setting can be transferred to "plain" BMDPs and, furthermore, to multi-objective MDPs even without introducing a probability distribution over the uncertainty set.

As we shall consider multi-objective optimization problems, we define the set of solutions that interest us.

*Definition 2.2 (**Pareto Optimality**).* Let $\mathcal{P}_{\text{pure}}$ be the set of pure policies $\pi\colon S \to A$, then we define the *Pareto frontier* as

$$\mathcal{P}_{\text{Pareto}} = \left\{\pi \in \mathcal{P}_{\text{pure}} \mid \nexists \pi' \in \mathcal{P}_{\text{pure}} : \mathbf{v}^{(\pi')} >_P \mathbf{v}^{(\pi)}\right\}$$

the set of pure policies that result in Pareto optimal value vectors. We denote by $\mathcal{V}_{\text{Pareto}}$ and $\mathcal{V}_{\text{pure}}$ the corresponding sets of value vectors.

It is easy to see that $\mathcal{P}_{\text{Pareto}} \subseteq \mathcal{P}_{\text{pure}}$ holds and usually, the subsets are proper. For convenience, we shall use an operator $PO(\cdot)$ that operates on sets with a partial order $\geq_P$ and is defined by $PO(X) = \left\{x \in X \mid \neg\exists y \in X : y >_P x\right\}$.

3 Computation of Optimal Policies

The computation of Pareto optimal policies in multi-objective MDPs is challenging as the number of optimal policies can be large or even infinite (for non-stationary policies). Therefore, algorithms are required to approximate the Pareto set efficiently instead of computing the whole set of Pareto optimal policies. Several works have already explored methods to compute or approximate the Pareto set, e.g., [1, 5, 14, 17, 26, 28] albeit for the computation of

Algorithm 1 Exact computation of $\mathcal{P}_{\text{Pareto}}$ and $\mathcal{V}_{\text{Pareto}}$

1: **function** PURE-OPT-EXACT ($\mathcal{P} = (S, A, T_\updownarrow, R_\updownarrow, Pr)$, γ)
2: $P \leftarrow \{\text{arbitrary policy}\}$ ▷ initialize current policy set
3: $F \leftarrow P$ ▷ initialize the Pareto frontier
4: **for** $i \in \{1, \ldots, |S|\}$ **do**
5: $P \leftarrow \cup_{\pi \in P} \left\{\pi' \mid d(\pi, \pi') = 1, \mathbf{v}^\pi \not\succ \mathbf{v}^{\pi'}\right\}$
6: $F \leftarrow PO(F \cup P)$
7: **return** F

several expected values or in a setting that has only a multi-objective reward. In SBMDPs we have to face optimization of, ultimately, several MDPs with related but not identical transition probabilities; furthermore, we consider the worst and best case vectors. This seems to be a different, and generally harder problem.

For most non-trivial models the number of Pareto optimal policies is still much too large to compute them all and many policies show a similar behavior. From a practical point of view it is sufficient to compute a subset of the Pareto frontier if the corresponding value vectors are evenly distributed in the value vector space. Theoretically, one can try value iteration or policy iteration-based approaches to compute policies from $\mathcal{P}_{\text{Pareto}}$ and the corresponding value vectors from $\mathcal{V}_{\text{Pareto}}$, as in [28]. The disadvantage of value iteration algorithms is that the number of intermediate partly evaluated policies can become enormous before the value iteration converges and the policies get completely evaluated. In this case, one has to stop the algorithm with a large number of non-comparable policies which might even not be optimal since value iteration provided only an approximation or a lower bound of the true value vectors. Thus, we consider a policy iteration-based approach.

In the following, we present two algorithms. The first algorithm computes $\mathcal{P}_{\text{Pareto}}$ exactly, albeit at a possibly high computational cost. The second algorithm is a fixed-point iteration that computes an increasing set of mutually non-dominating policies. Our first approach relies on the following result. The proofs of this and the following results can be found in the online companion [23].

LEMMA 3.1. *Let $\mathcal{P} = (S, A, T_\updownarrow, R_\updownarrow, Pr)$ be a SBMDP. Let furthermore π, π' be two policies where π' lies on the Pareto frontier. Then there exists a finite sequence of policies $\pi = \pi_0, \pi_1, \ldots, \pi_N = \pi'$ where $d(\pi_i, \pi_{i+1}) = 1, \mathbf{v}^{(\pi_i)} \not\succ \mathbf{v}^{(\pi_{i+1})}$ and, additionally, $N \leq |S|$.*

From this result, we develop a procedure that computes $\mathcal{P}_{\text{Pareto}}$ as sketched in Algorithm 1. Intuitively, the algorithm generates for each currently considered policy possible non-dominated neighbor policies and updates a global set of non-dominated policies. This continues for $|S|$ steps.

THEOREM 3.2. *Algorithm 1 correctly computes $\mathcal{P}_{\text{Pareto}}$.*

The complexity of the exact algorithm is, as of yet, unclear. In the worst possible case the algorithm will produce large numbers of temporarily optimal policies that will be dominated by a few Pareto-optimal policies in the end, making the worst-case complexity $O\left(m^n\right)$, theoretically independent of the final size of $\mathcal{P}_{\text{Pareto}}$. To avoid this, we propose a slightly different algorithm that computes a set of policies that seem to be a reasonably good approximation of $\mathcal{P}_{\text{Pareto}}$. It is important to note that the following approach is an heuristic; later, we shall discuss the (empirical) quality of this heuristic.

The heuristic approach is a simplification of the Algorithm 1. The simplification lies in including only those policies into the non-dominated set that are non-dominated with respect to all other policies. This decreases the number of candidate policies and, thus, the runtime at the cost of possible inaccuracy.

The practical implementation tries to find quickly policies that are spread over the Pareto frontier by analyzing first those policies which are likely to belong to the Pareto frontier. It is known that the policies $\pi_\downarrow, \overline{\pi}, \pi_\uparrow$ belong to $\mathcal{P}_{\text{Pareto}}$, as no other policy has a greater value vector in the lower bound, average, and upper bound case, respectively, by definition. Starting with these policies makes the algorithm walk through the policy space from the extreme points of the Pareto frontier which, as we hope, yields an evenly distributed non-dominated set of policies.

The following simple observation helps to find good policies which are candidates for the Pareto frontier without too many policy evaluations. Let for a policy π and a state-action pair (s, a) the *gradient* be

$$\nabla(\pi, s, a) = \left(r^a_{\downarrow s} + \gamma \min_{\mathbf{p}^a_s \in \mathbf{p}^a_{\updownarrow s}} \mathbf{p}^a_s \mathbf{v}^{(\pi)}_\downarrow - v^{(\pi)}_{\downarrow s}, \right. \\ \left. \overline{r}^a_s + \gamma \overline{\mathbf{p}}^a_s \overline{\mathbf{v}}^{(\pi)} - \overline{v}^{(\pi)}_s, r^a_{\uparrow s} + \gamma \max_{\mathbf{p}^a_s \in \mathbf{p}^a_{\updownarrow s}} \mathbf{p}^a_s \mathbf{v}^{(\pi)}_\uparrow - v^{(\pi)}_{\uparrow s} \right). \tag{3}$$

If, for some policy π, a state s and an action $a \in A \setminus \{\pi(s)\}$ can be found such that

$$\nabla(\pi, s, a) >_P \mathbf{0} \tag{4}$$

then a policy $\pi^{(s,a)}$ can be defined with $\pi^{(s,a)}(t) = \pi(t)$ for $t \neq s$ and $\pi^{(s,a)}(s) = a$ and for $\pi^{(s,a)}$ it is $\mathbf{v}^{(\pi^{(s,a)})} >_P \mathbf{v}^{(\pi)}$. We define an operator

$$\pi' = popt\left(\pi, \mathbf{v}^{(\pi)}\right)$$

that generates a new policy from π by selecting for each s the lexicographically smallest action a that observes (4), whenever this is possible. Looping *popt* until convergence, which occurs in the repeat-loop in lines 14–17, yields a locally optimal policy in the Pareto frontier.

The function *popt* considers only pairs (s, a) where the gradient is positive in all components, making $\pi^{(s,a)}$ dominate π. In fact, the gradient gives a hint how the new policy will behave. From the gradient, the direction of the different value vectors of a new policy can be estimated without evaluating it. Policy evaluation is done by a function *eval* which solves the following three sets of equations to compute the value vectors for some pure policy π.

$$\mathbf{u}_\downarrow = \mathbf{r}^{(\pi)}_\downarrow + \gamma \min_{P \in P^{(\pi)}_\updownarrow} \left(P\mathbf{u}_\downarrow\right), \overline{\mathbf{u}} = \overline{\mathbf{r}}^{(\pi)} + \gamma \overline{P}^{(\pi)} \overline{\mathbf{u}}, \\ \text{and } \mathbf{u}_\uparrow = \mathbf{r}^{(\pi)}_\uparrow + \gamma \max_{P \in P^{(\pi)}_\updownarrow} \left(P\mathbf{u}_\uparrow\right) \tag{5}$$

The equations for the average values define a set of linear equations which can be solved with standard techniques. For the vector of the worst and best case an iterative approach is applied: The vectors are successively computed by solving the LP and optimizing the matrix until convergence [8].

In our algorithm we use a set PV that contains tuples $(\pi, \mathbf{v}^{(\pi)})$. Additionally, we use a set P where all evaluated policies are stored

Algorithm 2 An heuristic for $\mathcal{P}_{\text{Pareto}}$ and $\mathcal{V}_{\text{Pareto}}$

function PURE-OPT($\mathcal{P} = (S, A, T_\updownarrow, R_\updownarrow, Pr)$, γ)
  $PV \leftarrow \{(\pi_\downarrow, \mathbf{v}_\downarrow), (\overline{\pi}, \overline{\mathbf{v}}), (\pi_\uparrow, \mathbf{v}_\uparrow)\}$;
  $P \leftarrow \{\pi_\downarrow, \overline{\pi}, \pi_\uparrow\}$;
  while $|PV| <$ max. number of policies **do**
    $\Pi = \emptyset$;
    for $\pi \in PV$ and all (s, a) where $\pi^{(s,a)} \notin P$ **do**
      $(g_\downarrow^{(\pi,s,a)}, \overline{g}^{(\pi,s,a)}, g_\uparrow^{(\pi,s,a)}) = \nabla(\pi, s, a)$; $\Pi = \Pi \cup \{(\pi, s, a)\}$;
    if no non-negative gradient exists for $(\pi, s, a) \in \Pi$ **then**
      break (all locally non-dominated solutions found);
    for $g \in \left\{g_\downarrow, \overline{g}, g_\uparrow\right\}$ **do**
      if $g^{(\pi,s,a)} > 0$ exists **then**
        $(\pi, s, a) \leftarrow \arg\max_{(\pi,s,a)\in\Pi} \left\{g^{(\pi,s,a)}\right\}$;
        $\pi' \leftarrow \pi^{(s,a)}$;
        repeat ▷ compute local optimum
          $\mathbf{v}^{(\pi')} = (\mathbf{v}_\downarrow, \overline{\mathbf{v}}, \mathbf{v}_\uparrow) \leftarrow eval(\mathcal{P}, \pi')$;
          $\pi' \leftarrow popt(\pi', \mathbf{v}^{(\pi')})$;
        until π' does not change
        $PV \leftarrow PO\left(PV \cup \{(\pi', \mathbf{v}^{(\pi')})\}\right)$; $P \leftarrow P \cup \{\pi'\}$;
    if PV was not changed **then**
      break (all new policies are explored or dominated)
  return PV

to avoid re-evaluations of policies. In the current description, new policies are generated starting from available policies by maximizing one direction of the gradient. The algorithm stops if in the current set of non-dominated policies, the neighbors of each policy are either explored or dominated. A further stopping condition is to explore a predefined number of policies.

As the algorithm is a heuristic, it is difficult to argue about guaranteed performance in terms of quality of the output, that is, if the resulting policies $P_r = \{\pi \mid (\pi, \cdot) \in PV\}$ fulfill $\mathcal{P}_{\text{Pareto}} \subseteq P_r$ and $P_r \subseteq \mathcal{P}_{\text{Pareto}}$. It is still possible to make several observations. The main difference between Alg. 2 and Alg. 1 lies in the bookkeeping that disallows Alg. 2 to explore policies that have been already evaluated and, more importantly, are dominated by some other policy that has been found. By doing this, Alg. 2 may ignore policies that are dominated yet lead (by choosing an appropriate sequence of policy changes) to the Pareto frontier; if there are no other ways to the Pareto frontier, this makes the output of Alg. 2 incomplete. On the other hand, one can provide a heuristic argument: Since the initial set of policies contains known "extreme points" $\pi_\downarrow, \overline{\pi}, \pi_\uparrow$, the policies that will be found by Alg. 2 in realistic settings will stem from a gradual transition from one extreme policy to another, as there will always exist a path of policies that improves one of the objectives until an optimum is reached. This way, we can expect that in real-life problems, the resulting set P_r will cover the Pareto frontier or at least the space between the value vector tuples $\mathbf{v}_\downarrow, \overline{\mathbf{v}}, \mathbf{v}_\uparrow$ adequately, i. e., the resulting set of value vectors will be evenly distributed in the space between the extreme value vectors stemming from $\pi_\downarrow, \overline{\pi}, \pi_\uparrow$. Furthermore, we expect that for practical problems, the following assumption will hold: If for a set of policies P it is $P \subseteq \mathcal{P}_{\text{Pareto}}$ and P contains not all Pareto optimal policies, then there exists a policy $\pi \in P$ and a state-action pair $(s, a) \in S \times A$ such that $\pi^{(s,a)} \notin P$ and $\pi^{(s,a)}$ is not dominated by any other policy in P. This especially means that there is always a "globally" non-dominated path of policies from a set P of mutually non-dominating policies to a policy in $\mathcal{P}_{\text{Pareto}} \setminus P$ if $\mathcal{P}_{\text{Pareto}} \neq P$. We expect that this assumption holds for practical instances. Furthermore, we conjecture that our assumption is also true for the problem in general. Finally, since Algorithm 2 is a variation of Algorithm 1, the same complexity reasoning applies here. However, the practical complexity should be lower than that of Algorithm 1, as less SBMDP evaluations have to be performed.

4 Evaluation

We have performed a series of experiments where we considered several questions. First, we compared the performance of Algorithm 2 against a black-box multi-objective optimization method as reference. We have chosen *SPEA2* [30] as reference since it is a well-studied, simple black-box optimization algorithm. Second, we have evaluated the general performance of the algorithm with respect to problem size and the number of computed solutions.

For the evaluation, we have used a machine with an Intel® Core™ i7-4790 CPU and 16 GB RAM. We have set a time limit of 1000 s for *SPEA2* and a limit for Algorithm 2 of 50 000 checked policies. The archive size for *SPEA2* has been set to 50 000. Concerning the implementations, we have used OpenMP parallelization methods to use multiple CPU cores when possible. Furthermore, we have used advanced numerical algorithms to evaluate (1) and (2). Specifically, for large instances, we substituted the direct LU solver [24] by preconditioned GMRES [18, 24] with an ILU0-preconditioner. The code and testing infrastructure are available at [22] and [21].

4.1 The Model of Multi-Server Queue

As the first case study, we have chosen a parameterizable model that can be easily generated; more concretely, we have considered a multi-server queue where servers can be switched off to save energy and switched on if the load in the system increases. Such queues are abstract models for server farms [7]. The goal is to find a compromise between small response times and low energy consumption.

We consider a system where customers arrive according to a Poisson process with rate λ and require an exponentially distributed service with mean μ^{-1}. As our algorithms are designed for discrete-time BMDPs, we consider a (morally) equivalent discrete model where the probability of arrival of a customer in a time unit is p, the service probability is q and, thus λ and μ are multiples of p^{-1} resp. q^{-1}. The system has a capacity of m and contains c servers. Each server can be in one of three states *on*, *off* and *start*. A server can be switched off after the end of a service or if it is idle. A server that is switched off immediately changes its state from *on* to *off*. Servers in state *off* can be switched on which means that they change their state to *start*. The duration of the starting period is exponentially distributed with rate ν, then the server changes its state to *on* and is ready to serve customers. A state of the system can be described by (i, j, k, l) where $i \in [0, m]$ describes the number of customers, j, k, l include the number of servers in state *on*, *start* and *off*, respectively. Consequently, $j + k + l = c$ has to hold. The number of states equals $n = (m + 1)(c + 2)(c + 1)/2$. The reward in the state (i, j, k, l) equals $(m - i)/(j\omega_1 + k\omega_2 + l\omega_3)$ where $\omega_1, \omega_2, \omega_3$ describe the energy consumption in the *on*, *start*, and *off* state.

The upper and lower bounds for the transition probabilities were chosen randomly by generating Gaussian noise and adding it to the transition probabilities defined above.

Comparison to a generic heuristic. As we consider a new problem, we have chosen to compare our approach to a black-box heuristic. More specifically, we have considered multi-objective optimization heuristics and decided to compare our algorithm to SPEA2 as it is a well-studied evolutionary optimization algorithm that is specifically designed to compute non-dominated sets for multi-objective optimization problems.

The SPEA2 algorithm. In detail, SPEA2 keeps two sets: a *population* P and an *archive* A where the non-dominated solutions are stored. In each iteration of the optimization cycle, A is updated with non-dominated elements of P. Then, a selection step takes place in which first, all elements of $A \cup P$ are assigned a *fitness value* and then, the solutions with lower fitness values are chosen to generate new solutions by application of mutation and crossover operators. The newly generated solutions are then the new population.

The distinctive feature of this approach is the fitness evaluation: The fitness of an individual solution p depends on the *strength* of other solutions p' that cover p. The strength itself is defined as the number of covered solutions; thus, the non-dominated solutions have maximal strength and minimal fitness values by definition, otherwise the ranking aims at picking more diverse solutions, i. e., solutions that are more evenly distributed in the objective value space.

In our SPEA2 implementation,we have defined problem-specific mutation and crossover operators. As possible solutions are stationary policies, the operators could be defined in a straightforward fashion. Mutation affects a decision in one state with probability $1/n$, if n is the number of states in the MDP, and replaces the previous action in the policy with a random one. Crossover takes two "parent" policies and chooses for the result an action from either of the original policies with probability $1/2$.

Comparison metrics. To quantify performance differences, we have used the coverage metric that has been introduced in [31]. This performance metric has been designed to compare two output sets of (heuristic) multi-objective optimization algorithms on the same problem and computes the fraction of one output that is covered (i. e., is dominated by or equal to) by an element of the other output set. Concretely, for two sets of vectors X and Y, the coverage metric is defined by $C(X,Y) = \frac{1}{|Y|}\left|\left\{y \in Y \mid \exists x \in X : x \geq y\right\}\right|$. $C(X,Y) = 1$ means that all points in Y are dominated by or equal to points in X whereas $C(X,Y) = 0$ means that no point in Y is covered by a point in X. It is worth noting that the coverage metric is asymmetric and in most cases it is interesting to compute both $C(X,Y)$ and $C(Y,X)$.

Results. The results of the comparison can be found in Fig. 1. The first figure describes the coverage metric where the first argument is the policy set computed by Alg. 2, the second figure describes the converse. In the run, *SPEA2* took always 1000 s while Alg. 2 did never take more than 330 s.

It is easy to see that Algorithm 2 almost always delivers a significantly better performance with respect to both time complexity as well as quality of computed policies. More concretely, in nearly all cases Algorithm 2 has computed a set that completely covered all solutions generated by *SPEA2*; the evolutionary heuristic, however, has never produced a policy that strictly dominated a policy from Alg. 2 and was only able to yield comparable solutions on small instances with state space size of at most 20.

Comparison to an exact computation. For some instances, we have furthermore compared the performance of Algorithm 2 to the exact approach in Algorithm 1. Concretely, we have considered the case $m = 2, c = 3$. It has turned out that for this case, the coverage metric was always 1. This suggests that Algorithm 2 may compute the complete Pareto frontier not only heuristically but also in theory. This is, however, a conjecture subject to further investigation.

4.2 Time complexity measurements

As the number of policies was bounded by an upper limit of 50 000 and $|A| = o(|S|)$, the complexity can be roughly estimated by a cubic term in $|S|$. For practical applications, we are also interested in runtimes on real-life instances. For this, we have used a slightly different but more general (and somewhat more scalable) model.

The grid model. We have chosen a model that resembles a grid with nm states $S = \left\{s_{i,j} \mid i \in [n], j \in [m]\right\}$ and m actions $A = [m]$. The rewards for actions in each state are chosen randomly with mean 100 and variance 20. The transition probabilities are also chosen randomly according to the Dirichlet distribution. Concretely, the transition probability vector from state $s_{i,j}$ to states $s_{\min(n,i+1),j'}$ for action a is Dirichlet-distributed with concentration parameters $\left(\alpha^a_1, \ldots, \alpha^a_m\right)$ where $\alpha^a_{j'} = 10$ if $a = j'$ and $\alpha^a_{j'} = 1$ otherwise which yields an (expected) 10 times larger probability to land in $s_{i+1,a}$ than in other states. The upper and lower bounds are, as before, generated by adding Gaussian noise.

Complexity. We present the results in graphical form. To get the results, we have run Algorithm 2 on instances with up to 400 states, with n and m between 5 and 20. For each (n,m), we have created 4 instances.

The results can be seen in Fig. 2a and Fig. 2b. The red dots are the empirical data, the blue lines describe a confidence interval that stems from a (scaled) t-distribution guess. The green line is the cubic regression term for convenience. One can see that the experimental performance generally matches a polynomial term, but also that even on large instances, the mean time until a non-dominated solution is generated lies under a second.

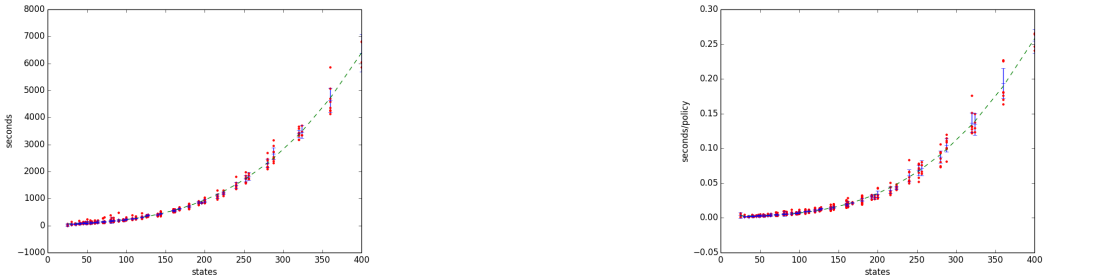
4.3 The Model of Autonomous Nondeterministic Tour Guides

Our second case study is inspired by "Autonomous Nondeterministic Tour Guides" (ANTG) in [4, 10], which models a complex museum with a variety of collections.

Models in [4] are MDPs. In our experiment, we will insert some uncertainties into the MDP. Due to the popularity of the museum, there are many visitors at the same time. Different visitors may have different preferences of arts. We assume the museum divides all collections into different categories so that visitors can choose what they would like to visit and pay tickets according to their preferences. In order to obtain the best experience, a visitor can first assign certain weights to all categories denoting their preferences to the museum, and then design the best strategy for a target. However,

(a) C_{IE} **(b)** C_{EI}

Figure 1: The C-measures in dependence of state space size



(a) Total time in dependence of problem size **(b) Mean time for a policy in dependence of problem size**

Figure 2: Time complexity measurements on the grid model

the preference of a sort of arts to a visitor may depend on many factors like price, weather, or the length of queue at that moment etc., hence it is hard to assign fixed values to these preferences. In our model we allow uncertainties of preferences such that their values may lie in an interval.

For simplicity we assume all collections are organized in an $n \times n$ square with $n \geq 10$. Let $m = \frac{n-1}{2}$. We assume all collections at (i,j) are assigned with a weight 1 if $|i-m| > \frac{n}{5}$ or $|j-m| > \frac{n}{5}$, with a weight 2 if $|i-m| \in (\frac{n}{10}, \frac{n}{5}]$ or $|j-m| \in (\frac{n}{10}, \frac{n}{5}]$; otherwise they are assigned with a weight interval $[3,4]$. In other words, we expect collections in the middle will be more popular and subject to more uncertainties than others. Furthermore, we assume that people at each location (i,j) have two non-deterministic choices: either move to the north and east, that is, $(\min(n, i+1), \min(n, j+1))$ or to the north and west, that is, $(\min(n, i+1), \max(0, j-1))$ if $i \geq j$, while if $i \leq j$, they can move either to the north and east, $(\min(n, i+1), \min(n, j+1))$, or to the south and east, that is, $(\max(0, i-1), \min(n, j+1))$. The transitions also depend on the location of the collection. For the collections in the middle, the main direction of transition is chosen with probability $[0.8, 1]$ while the probability to move to some other neighbor collection is $[0, 0.2]$. In the expected case, we set the probability to move to the collection in the main direction to 0.8 and distribute the remaining probability mass evenly among other neighbor collections. For collections outside the middle, the main direction (for example, north and west) is chosen with probability 1.

Therefore a model with parameter n has n^2 states in total and roughly $2n^2$ transitions, 2% of which are associated with uncertain weights and uncertain transition probabilities. Notice that a transition with uncertain weights essentially corresponds to several transitions with concrete weights.

We define a reward structure denoting the reward one can obtain by visiting each collection. For simplicity, we let the reward be the same as the weight of a collection. We can ask for the optimal policy for the expected discounted reward criterion, that is, in the scenario where it is preferable to make better rewarding moves early.

Evaluation of the ANTG model We have performed an evaluation of the model for $10 \leq n \leq 20$, with the results depicted in Fig. 3. For convenience, the runtime and the number of policies are plotted in dependence of the number of states, which is n^2. We see that on large instances, the problem structure yields a large number of optimal policies, thus decreasing performance. On small instances, however, the number of optimal policies generated is small, which allows for a fast computation of the Pareto frontier.

5 Conclusions

In this paper, we have presented a novel approach to analyze bounded-parameter Markov decision processes. In contrast to known approaches [8] that usually analyze the worst case behavior and result in a variant of robust optimization, the problem is handled here as a multi-objective problem. Of course, the price for this extension can be an exponential complexity due to an exponential

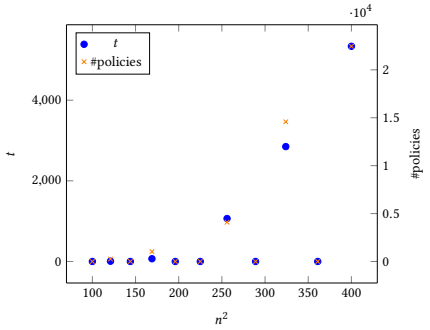


Figure 3: Evaluation of the ANTG model

number of *optimal* and mutually incomparable policies in this case. The problem differs from many other multi-objective optimization approaches for MDPs where several expected values are analyzed, whereas here worst, expected and best case behavior are considered together which together yields a *multi-scenario* optimization problem.

In order to find all Pareto optimal policies in multi-objective optimization of BMDPs, two computational algorithms are presented. One of the algorithms computes the desired set of policies exactly, but has a prohibitive runtime complexity. The second algorithm that is explicitly designed for the problem is a heuristic and seems to perform well on a large number of test instances. In particular, we have shown that the policies obtained by the heuristic were in almost all cases better than those produced by a generic evolutionary multi-objective optimization algorithm.

The approach presented here can be extended in various directions. As mentioned it is fairly easy to consider additional goals beyond worst, average and best cases. Our results not only apply to bounded-parameter Markov decision processes, but can also be utilized for Markov decision processes with convex uncertainties [15]. Therefore, the same basic algorithms can be used in order to compute Pareto optimal policies, if one adjusts them to solve the convex program in the iteration step.

For future research, it would be interesting to consider the stochastic multi-scenario problem for stochastic BMDPs. There, the goal is to optimize the expected value of the value vector, in contrast to the value vector of the expected MDP we have used here. This problem is a slightly different, single-objective optimization problem and needs to be considered separately.

Acknowledgments

This work is supported by the DFG as part of RTG 1855 and by the ERC Advanced Grant 695614 (POWVER).

References

[1] Leon Barrett and Srini Narayanan. 2008. Learning All Optimal Policies with Multiple Criteria. In *ICML*. 41–47.
[2] Dimitris Bertsimas and Velibor V. Mišić. 2017. Robust Product Line Design. *Operations Research* 65, 1 (2017), 19–37. https://doi.org/10.1287/opre.2016.1546
[3] Dimitris Bertsimas, John Silberholz, and Thomas Trikalinos. 2016. Optimal healthcare decision making under multiple mathematical models: application in prostate cancer screening. *Health care management science* (2016), 1–14.
[4] Andrew S. Cantino, David L. Roberts, and Charles L. Isbell. 2007. Autonomous nondeterministic tour guides: improving quality of experience with TTD-MDPs. In *AAMAS*. IFAAMAS, 22.
[5] Krishnendu Chatterjee, Rupak Majumdar, and Thomas A. Henzinger. 2006. Markov Decision Processes with Multiple Objectives. In *STACS*. 325–336.
[6] Karina Valdivia Delgado, Scott Sanner, and Leliane Nunes de Barros. 2011. Efficient solutions to factored MDPs with imprecise transition probabilities. *Artif. Intell.* 175, 9-10 (2011), 1498–1527.
[7] Anshul Gandhi, Mor Harchol-Balter, and Ivo J. B. F. Adan. 2010. Server farms with setup costs. *Perform. Eval.* 67, 11 (2010), 1123–1138.
[8] Robert Givan, Sonia M. Leach, and Thomas L. Dean. 2000. Bounded-parameter Markov decision processes. *Artif. Intell.* 122, 1-2 (2000), 71–109.
[9] Ernst Moritz Hahn, Vahid Hashemi, Holger Hermanns, Morteza Lahijanian, and Andrea Turrini. 2017. Multi-objective Robust Strategy Synthesis for Interval Markov Decision Processes. *CoRR* abs/1706.06875 (2017). http://arxiv.org/abs/1706.06875
[10] Vahid Hashemi, Holger Hermanns, and Lei Song. 2016. Reward-Bounded Reachability Probability for Uncertain Weighted MDPs. In *International Conference on Verification, Model Checking, and Abstract Interpretation*. Springer, 351–371.
[11] Garud N. Iyengar. 2005. Robust Dynamic Programming. *Mathematics of Operations Research* 30, 2 (2005), 257–280.
[12] K. Klamroth, E. Köbls, A. Schöbel, and Chr. Tammer. 2013. A unified approach for different concepts of robustness and stochastic programming via non-linear scalarizing functionals. *Optimization* 62, 5 (2013), 649–671.
[13] Andrey Kolobov. 2012. Planning with Markov decision processes: An AI perspective. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 6, 1 (2012), 1–210.
[14] Patrice Perny and Paul Weng. 2010. On Finding Compromise Solutions in Multiobjective Markov Decision Processes. In *ECAI*. 969–970.
[15] Alberto Puggelli, Wenchao Li, Alberto L. Sangiovanni-Vincentelli, and Sanjit A. Seshia. 2013. Polynomial-Time Verification of PCTL Properties of MDPs with Convex Uncertainties. In *CAV*. 527–542.
[16] Martin L. Puterman. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA.
[17] Diederik M. Roijers, Shimon Whiteson, and Frans A. Oliehoek. 2014. Linear Support for Multi-objective Coordination Graphs. In *AAMAS*. 1297–1304.
[18] Youcef Saad. 1993. A Flexible Inner-Outer Preconditioned GMRES Algorithm. *SIAM J. Scientific Computing* 14, 2 (1993), 461–469.
[19] Jay K. Satia and Roy E. Lave. 1973. Markovian Decision Processes with Uncertain Transition Probabilities. *Operations Research* 21, 3 (1973), 728–740.
[20] Dimitri Scheftelowitsch. 2015. The Complexity of Uncertainty in Markov Decision Processes. In *SIAM Conference on Control & its Applications CT15*.
[21] Dimitri Scheftelowitsch. 2016. BMDP testing infrastructure. https://gitlab.com/dreval/bmdp-python-scripts. (2016). Accessed: 2016-11-22.
[22] Dimitri Scheftelowitsch and Peter Buchholz. 2016. BMDP analysis tool. https://gitlab.com/dreval/bmdp-analysis. (2016). Accessed: 2017-07-17.
[23] Dimitri Scheftelowitsch, Peter Buchholz, Vahid Hashemi, and Holger Hermanns. 2017. Online Companion for "Multi-Objective Approaches to Markov Decision Processes with Uncertain Transition Parameters". http://ls4-www.cs.tu-dortmund.de/cms/de/home/scheftelowitsch/publications/bmdp-mo-online-companion.pdf. (2017).
[24] William J. Stewart. 1994. *Introduction to the numerical solution of Markov Chains*. Princeton University Press.
[25] Chelsea C. White and Hany K. Eldeib. 1994. Markov Decision Processes with Imprecise Transition Probabilities. *Operations Research* 42, 4 (1994), 739–749. https://doi.org/10.1287/opre.42.4.739 arXiv:http://dx.doi.org/10.1287/opre.42.4.739
[26] DJ White. 1982. Multi-objective infinite-horizon discounted Markov decision processes. *Journal of mathematical analysis and applications* 89, 2 (1982), 639–647.
[27] Margaret M Wiecek, Vincent Y Blouin, Georges M Fadel, Alexander Engau, Brian J Hunt, and Vijay Singh. 2009. Multi-scenario multi-objective optimization with applications in engineering design. In *Multiobjective Programming and Goal Programming*. Springer, 283–298.
[28] M.A. Wiering and E.D. de Jong. 2007. Computing Optimal Stationary Policies for Multi-Objective Markov Decision Processes. In *ADPRL 2007*. 158–165.
[29] Wolfram Wiesemann, Daniel Kuhn, and Berç Rustem. 2013. Robust Markov Decision Processes. *Math. Oper. Res.* 38, 1 (2013), 153–183.
[30] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. 2001. *SPEA2: Improving the Strength Pareto Evolutionary Algorithm*. Technical Report 103. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.17.2440
[31] Eckart Zitzler and Lothar Thiele. 1999. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. *IEEE Trans. Evolutionary Computation* 3, 4 (1999), 257–271.