Solving Robust Markov Decision Processes: Generic, Reliable, Efficient

Tobias Meggendorfer[1*], **Maximilian Weininger**[2*], **Patrick Wienhöft**[3,4*]

[1]Lancaster University Leipzig, Leipzig, Germany
[2]Institute of Science and Technology Austria, Klosterneuburg, Austria
[3]Dresden University of Technology, Dresden, Germany
[4]Centre for Tactile Internet with Human-in-the-Loop (CeTI), Dresden, Germany
tobias@meggendorfer.de, maximilian.weininger@ista.ac.at, patrick.wienhoeft@tu-dresden.de

Abstract

Markov decision processes (MDP) are a well-established model for sequential decision-making in the presence of probabilities. In *robust* MDP (RMDP), every action is associated with an *uncertainty set* of probability distributions, modelling that transition probabilities are not known precisely. Based on the known theoretical connection to stochastic games, we provide a framework for solving RMDPs that is generic, reliable, and efficient. It is *generic* both with respect to the model, allowing for a wide range of uncertainty sets, including but not limited to intervals, L^1- or L^2-balls, and polytopes; and with respect to the objective, including long-run average reward, undiscounted total reward, and stochastic shortest path. It is *reliable*, as our approach not only converges in the limit, but provides precision guarantees at any time during the computation. It is *efficient* because – in contrast to state-of-the-art approaches – it avoids explicitly constructing the underlying stochastic game. Consequently, our prototype implementation outperforms existing tools by several orders of magnitude and can solve RMDPs with a million states in under a minute.

Code — https://zenodo.org/records/14385450
Extended version — https://arxiv.org/abs/2412.10185

1 Introduction

Robust Markov decision processes. *Markov decision processes (MDPs)* (Puterman 1994) are *the* standard model for sequential decision making and planning in the context of non-determinism and uncertainty. In brief, an MDP proceeds as follows: Starting in some state of the modelled system, an agent chooses an action (resolving non-determinism) and the MDP continues to a successor state, sampled from a probability distribution associated with the state-action pair (resolving uncertainty). In practice, this uncertainty is often not known precisely, but rather estimated from data. *Robust Markov decision processes (RMDPs)* (Nilim and Ghaoui 2005; Iyengar 2005) are an extension of MDPs that lift the assumption of knowing every transition probability exactly. Instead of one precise probability distribution per state-action pair, RMDPs have an *uncertainty set* consisting of (potentially uncountably) many probability distributions. RMDPs have been used in, e.g., healthcare applications (Zhang, Steimle, and Denton 2017). However, existing approaches to solving RMDP suffer from significant drawbacks, e.g. they are limited to very specific objectives or classes of uncertainty sets, or do not provide any guarantees on the correctness of their result. Alleviating all these problems, we present a framework for solving RMDPs that is *generic*, *reliable*, and *efficient*.

Generic Uncertainty Sets. In the literature, many variants of uncertainty sets exist: Firstly, polytopic uncertainty sets (Chatterjee et al. 2024) generalize simple interval uncertainty, e.g. (Givan, Leach, and Dean 2000; Tewari and Bartlett 2007), L^1-balls around a given probability distribution, e.g. (Strehl and Littman 2004; Ho, Petrik, and Wiesemann 2018), and contamination models (Wang et al. 2024). Definable uncertainty sets (Grand-Clément, Petrik, and Vieille 2023) are a recent generalization of polytopic uncertainty. Secondly, non-polytopic (and non-definable) uncertainty sets include the Chi-square (Iyengar 2005), Kullback-Leibler divergence (Nilim and Ghaoui 2005), and Wasserstein distance (Yang 2017) uncertainty sets, and L^p-balls around a distribution for $1 < p < \infty$, all of which state-of-the-art methods cannot handle in general. In this paper, we introduce the *Constant-Support* Assumption, which intuitively requires that the successors of an action are certain, and only the transition probabilities are unknown. It allows us to capture all the listed non-polytopic variants and more. *Constant-Support* uncertainty sets are incomparable to both polytopic and definable uncertainty sets, i.e. there exists polytopic (and definable) uncertainty sets that do not satisfy the *Constant-Support* Assumption and vice versa.

Solution algorithms are always restricted to some particular representation of uncertainty. Considering more general uncertainty sets comes with several complications, e.g. that in some cases optimal policies may cease to exist (see Ex. 1). In this work, we consider a large class of uncertainty sets by investigating ones that are polytopic or that satisfy the *Constant-Support* Assumption.

Generic Objectives. RMDPs mainly have been investigated with *discounted* or *finite-horizon objectives*, which put an emphasis on the immediate performance of the system, see e.g. the seminal works (Nilim and Ghaoui 2005; Iyengar 2005) or the recent overview (Wang et al. 2024, Sec. 1.2). While these objectives can be included in our framework,

[*]These authors contributed equally.
Copyright © 2025, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

they are not the focus of this paper and are only discussed in the extended version of the paper (Meggendorfer, Weininger, and Wienhöft 2024, App. B). Recently, several works studied RMDPs with *long-run average reward (LRA)* objectives (Chatterjee et al. 2023; Wang et al. 2024). We discuss these and their relation to our framework in the related work section below. Lastly, the popular objectives of *undiscounted total reward (TR)* and *stochastic shortest path (SSP)* have only been investigated in the very restricted setting of interval uncertainty sets (Buffet 2005; Wu and Koutsoukos 2008), while we provide solutions for the more general RMDP setting. The importance of RMDPs with these objectives for several research fields is discussed in (Badings et al. 2023).

Technical Contribution. To achieve generality, reliability, and efficiency, we exploit two key ideas. Firstly, RMDPs can be reduced to *stochastic games (SGs)*, where one player is the agent of the RMDP, and the other player is the environment, picking a probability distribution from the uncertainty set. This connection was mentioned already in (Nilim and Ghaoui 2005; Iyengar 2005) for finite horizon and discounted objectives, and recently was formalized for polytopic uncertainty sets and LRA objectives (Chatterjee et al. 2024). We extend this reduction to arbitrary uncertainty sets, as well as to TR and SSP objectives. Thus, we can apply *theoretical* results from SG literature to tackle RMDPs reliably.

Secondly, previous works exploiting this connection either do not address how to *practically* find the decisions of the other player (Nilim and Ghaoui 2005; Iyengar 2005), or explicitly construct the induced SG (Chatterjee et al. 2024). The latter approach is not applicable to general uncertainty sets, as the SG can be infinite; and for polytopic uncertainty sets, while finite, the SG requires exponential space. Here, we show how this explicit construction can be avoided, performing the key steps of the SG solution algorithm implicitly. This mitigates both drawbacks of the explicit approach.

Reliable Stopping Criterion. Several approaches for solving RMDPs are based on value iteration (VI), e.g. (Grand-Clément, Petrik, and Vieille 2023; Wang et al. 2024). However, these only converge in the limit and cannot bound the current imprecision. Consequently, in practice they are terminated when the result can still be arbitrarily far off and unreliable. The problem of obtaining guarantees on the precision and a stopping criterion has been a major area of research in the past decade for non-robust systems, see e.g. (Brázdil et al. 2014; Baier et al. 2017; Haddad and Monmege 2018) or (Kretínský, Meggendorfer, and Weininger 2023a) for a unified framework subsuming MDPs and SGs with quantitative objectives. We extend these new advances to RMDPs.

Efficient through Implicit Updates. While this extension is straightforward in theory using the explicit reduction to SGs, we provide an implicit approach, addressing a major technical challenge. These implicit updates not only avoid the exponential space requirement, but for many practically relevant uncertainty sets can be computed in polynomial time. Our experimental evaluation shows that in practice this results in several orders of magnitude improvements over approaches that explicitly construct the induced SG. In particular, we not only consider small, handcrafted examples from previous works, but also use MDPs from well-established benchmark sets (Hartmanns et al. 2019) and complement them with uncertainty sets, thus demonstrating that our approach scales to RMDPs with complex structure and millions of states.

Algorithm Overview. Firstly, we provide an efficient implicit algorithm applicable to a wide range of RMDP that converges in the limit but does not give guarantees on its result (Alg. 1). For RMDPs satisfying the *Constant-Support* Assumption, we provide algorithms with guaranteed stopping criterion that are implicit and thus efficient for TR and SSP (Alg. 2), and for long-run average reward (Alg. 3 in (Meggendorfer, Weininger, and Wienhöft 2024, App. F)). For polytopic RMDPs violating *Constant-Support*, we provide an implicit algorithm with stopping criterion for maximizing TR and SSP (see paragraph "Beyond *Constant-Support*" at the end of Section 5). For the other objectives (minimizing TR and LRA) in polytopic RMDPs, we either offer the aforementioned implicit and convergent Alg. 1 without reliable stopping criterion, or an algorithm with guaranteed stopping criterion that is explicit and hence less efficient (building on Thm. 1). We highlight that if convergence in the limit is sufficient and no sound stopping criterion is required, Alg. 1 provides the most efficient solution that works for all considered objectives and uncertainty sets.

Summary. By deepening the understanding of the connection between RMDPs and SGs and exploiting recent advances on SG solving, we obtain a generic framework able to deal with more variants of uncertainty sets and objectives than the state-of-the-art. At the same time, our approach provides a correct stopping criterion, ensuring reliability, and, through implicit computation, is orders of magnitude more efficient than existing, explicit solution approaches.

Related Work. In (Wang et al. 2023) (journal version (Wang et al. 2024)) the authors provide a *value iteration (VI)* solution to the LRA objective. However, they provide no stopping criterion and restrict the graph structure of the RMDPs to be unichain. In (Chatterjee et al. 2024), the authors provide a complexity analysis and policy iteration algorithm for the same problem. They only consider polytopic uncertainty sets and explicitly construct the induced SG, resulting in the exponential space requirement. Finally, (Grand-Clément, Petrik, and Vieille 2023) proposes value iteration algorithms similar to ours, but leave providing a stopping criterion as an open question. Moreover, in their practical implementation, they significantly restrict the uncertainty sets. When restricting to *interval* RMDPs, `PRISM` (Kwiatkowska, Norman, and Parker 2011) supports TR objectives and `IntervalMDP.jl` (Mathiesen, Lahijanian, and Laurenti 2024) supports reachability and discounted rewards, focussing on parallelization. However, both tools offer no guarantees, employing an unsound stopping criterion. Our experimental evaluation (Section 6) compares with all mentioned related works except (Grand-Clément, Petrik, and Vieille 2023) which does not provide an implementation.

2 Preliminaries

A *probability distribution* over a finite or countable set X is a mapping $d : X \to [0,1]$, such that $\sum_{x \in X} d(x) = 1$. The set of all probability distributions on X is $\mathcal{D}(X)$. We denote the support of a probability distribution $p \in \mathcal{D}(X)$ by $\mathrm{supp}(p) = \{x \in X \mid p(x) > 0\}$.

Markov Decision Process. A (finite-state, discrete-time) *Markov decision process (MDP)*, e.g. (Puterman 1994), is a tuple $\mathsf{M} = (S, A, \mathsf{P}, \mathsf{r})$, where S is a finite set of *states*; A is a finite set of *actions*; $\mathsf{P}\colon S \times A \rightharpoonup \mathcal{D}(S)$ is a (partial) *transition function* mapping state-action pairs to a distribution over successor states; and $\mathsf{r}\colon S \times A \to \mathbb{N}$ is a *reward function* mapping state-action pairs to non-negative rewards (see (Meggendorfer, Weininger, and Wienhöft 2024, App. A) for a reduction from commonly occurring reward functions to natural numbers). We denote by $A(s) \neq \emptyset$ the *available* actions of a state s where $\mathsf{P}(s,a)$ is defined.

The semantics of MDPs are defined by means of *policies* which are mappings from finite paths (also called histories) to distributions over actions, formally $(S \times A)^* \times S \to \mathcal{D}(A)$. Intuitively, a path in an MDP initially consists only of some state s, and evolves under a policy π by sampling an action a according to the distribution $\pi(s)$, receiving the reward $\mathsf{r}(s,a)$, and transitioning to the next state s' sampled according to $\mathsf{P}(s,a)$. The process continues in this way ad infinitum, always using the whole path as input for the policy (e.g. $\pi(sas')$ in the second step). A policy is *memoryless* if it only depends on the current state and *deterministic* if it assigns probability 1 to a single action.

Robust MDP. A *robust MDP (RMDP)* $\mathcal{M} = (S, A, \mathcal{P}, \mathsf{r})$ (Nilim and Ghaoui 2005) is a generalization of MDPs, where instead of a fixed distribution the transition function yields an *uncertainty set* $\mathcal{P}(s,a)$. More formally, $\mathcal{P}\colon S \times A \to 2^{\mathcal{D}(S)}$, where 2^X denotes the set of all subsets of X. We say that an RMDP is *closed* if $\mathcal{P}(s,a)$ is closed for every state-action pair. We employ the classical assumption that uncertainty sets are (s,a)-rectangular as in, e.g. (Nilim and Ghaoui 2005; Chatterjee et al. 2024; Wang et al. 2024), i.e. the uncertainty sets are independent for each state-action pair. Intuitively, there is one additional step in the evolution of an RMDP: Before the successor state is sampled, the environment chooses a distribution from the uncertainty set $\mathcal{P}(s,a)$ according to an *environment policy* τ, i.e. $\tau(s_0 a_0 \dots s_n a_n) \in \mathcal{P}(s_n, a_n)$. Formally, a pair of policies π for the agent and τ for the environment induces a probability measure $\mathbb{P}^{\pi,\tau}_{\mathcal{M}}$ over infinite paths in an RMDP $\mathcal{M}$, see (Meggendorfer, Weininger, and Wienhöft 2024, App. A) for details. We denote by $\mathbb{E}^{\pi,\tau}_{\mathcal{M},s}$ the expectation under this probability measure when using s as initial state.

Objectives. Objectives define a mapping from infinite paths $\rho = s_0 a_0 s_1 a_1 \cdots$ to their payoff. We consider undiscounted total reward (TR) as well as long-run average reward (LRA) (Puterman 1994, Chps. 7 & 8), where

$$\mathsf{TR}(\rho) = \sum\nolimits_{t=0}^{\infty} \mathsf{r}(s_t, a_t) \text{ and}$$

$$\mathsf{LRA}(\rho) = \liminf_{n\to\infty} \frac{1}{n} \sum\nolimits_{t=0}^{n-1} \mathsf{r}(s_t, a_t).$$

(Further details can be found in (Meggendorfer, Weininger, and Wienhöft 2024, App. A), in particular how *stochastic shortest path (SSP)* is a variant of TR.) Using $\mathsf{Payoff} \in \{\mathsf{TR}, \mathsf{LRA}\}$, the *value* of a state s in an RMDP $\mathcal{M}$ under policies (π, τ) is

$$\mathsf{V}^{\pi,\tau}_{\mathcal{M}}(s) = \mathbb{E}^{\pi,\tau}_{\mathcal{M},s}[\mathsf{Payoff}].$$

Problem Statement. The *optimal value of an RMDP* $\mathcal{M}$ is the value under the best possible policy of the agent in whichever instantiation the environment chooses. We consider both the problems of maximizing or minimizing the payoff (interpreting rewards as costs when minimizing), defined by

$$\mathsf{V}^{\max}_{\mathcal{M}}(s) = \sup_{\pi} \inf_{\tau} \mathsf{V}^{\pi,\tau}_{\mathcal{M}}(s) \text{ and } \mathsf{V}^{\min}_{\mathcal{M}}(s) = \inf_{\pi} \sup_{\tau} \mathsf{V}^{\pi,\tau}_{\mathcal{M}}(s)$$

We often use the shorthands $opt \in \{\max, \min\}$ and V^{opt} to talk about both maximizing and minimizing objectives. For simplicity, we write $\overline{opt} = \max$ if $opt = \min$ and $\overline{opt} = \min$ if $opt = \max$.

Uncertainty Set Variants. Throughout the paper, we mostly distinguish two different variants of uncertainty sets: polytopic and arbitrary.

Definition 1. *We say an RMDP $\mathcal{M} = (S, A, \mathcal{P}, \mathsf{r})$ is* polytopic *if for each state-action pair (s,a) the uncertainty set $\mathcal{P}(s,a) \subseteq \mathbb{R}^{|S|}$ is a* polytope.

A polytope is the convex hull of finitely many points $\mathbb{R}^{|S|}$ or, equivalently, it is the intersection of a finite family of closed half-spaces (Grünbaum 2003). Thus, a polytope classically is given in $\mathcal{V}$-representation (vertex) or $\mathcal{H}$-representation (half-space). Computing the $\mathcal{V}$-representation from $\mathcal{H}$-representation can result in exponentially many vertices, e.g. already when an interval on every probability is given (Sen, Viswanathan, and Agha 2006, Lem. 6). Many uncertainty set variants are polytopic, for example L^1-balls around a probability distribution. However, e.g. L^2-balls are not polytopes, which brings us to the second variant: In *arbitrary* RMDPs, the uncertainty sets are not restricted at all. For our algorithmic results, we assume that the uncertainty sets are closed and satisfy the *Constant-Support* Assumption, explained in Sec. 3.

RMDPs Semantics. There are several semantics for RMDP, related to how probability distributions are chosen from the uncertainty set, see e.g. (Nilim and Ghaoui 2005; Iyengar 2005). The main questions are (i) "Is the environment choosing the distribution allowed to use memory (time-varying) or not (stationary)?", and (ii) "Is the environment an ally (best-case) or an antagonist (worst-case)?". We prove that for a large class of RMDPs, (i) is irrelevant, as the environment has equal power in both cases (see Cors. 1 and 2). As to (ii), our solutions for the harder worst-case are also applicable to the best-case (see (Meggendorfer, Weininger, and Wienhöft 2024, App. A) for further discussion). Note that if we assume a stochastic environment instead of an ally or antagonistic environment, the RMDP can be reduced to a standard MDP by collapsing the stochasticity of the environment and the system into a single step.

Turn-based Stochastic Game. Our solution approach utilizes a reduction to *turn-based stochastic games (SG)*. A finite-action SG (Condon 1992; Chen et al. 2013) is a tuple $\mathcal{G} = (S, A, \mathsf{P}, \mathsf{r})$ where all components are as for MDP, but with an additional partitioning of S into *max-states* S_{max} and *min-states* S_{min}. Additionally, we define *infinite-action SGs* as SGs where we allow the set of actions A to be infinite. We distinguish between (finite) max-paths and min-paths depending on whether the last state in the path is a max- or a min-state. The semantics of a SG are determined by a pair of policies π, τ, one for each player, with the max-player deciding the next action for a max-path and the min-player for min-paths. These induce a probability measure over infinite paths, which is used to compute the expectation of the given objective. Using $\mathsf{V}^{\pi,\tau}_{\mathcal{G}}(s) = \mathbb{E}^{\pi,\tau}_{\mathcal{G},s}[\mathsf{Payoff}]$, the value of an SG $\mathcal{G}$ is defined analogously to RMDPs, i.e. $\mathsf{V}^{\max}_{\mathcal{G}}(s) = \sup_\pi \inf_\tau \mathsf{V}^{\pi,\tau}_{\mathcal{G}}(s)$ and $\mathsf{V}^{\min}_{\mathcal{G}}(s) = \inf_\pi \sup_\tau \mathsf{V}^{\pi,\tau}_{\mathcal{G}}(s)$

3 Connection Between RMDP and SG

Since the environment in an RMDP acts as an antagonist to the agent, there is a natural correspondence between RMDP and SG, as noted in e.g. (Nilim and Ghaoui 2005). Intuitively, we can add a second player who chooses the instance of the RMDP. This player then chooses one action from the uncertainty set at every state. Thus, we alternate between original MDP state s where the optimizing player chooses action a, and a new antagonistic state s^a where the environment selects some probability distribution from the uncertainty set, see, e.g., (Iyengar 2005; Nilim and Ghaoui 2005; Chatterjee et al. 2024). For the formal definition, recall that $opt \in \{\max, \min\}$ denotes the optimization direction of the agent's objective and $\overline{opt}$ is the environment's optimization direction, i.e. the "inverse" of opt.

Definition 2 (Induced SG for arbitrary RMDP)**.** *For an arbitrary RMDP $\mathcal{M} = (S, A, \mathcal{P}, \mathsf{r})$ with an opt-objective, its induced SG $\mathcal{G}_{\mathcal{M}} = (S^{\mathcal{G}}, A^{\mathcal{G}}, \mathsf{P}^{\mathcal{G}}, \mathsf{r}^{\mathcal{G}})$ is defined as follows:*

- $S^{\mathcal{G}} = S^{\mathcal{G}}_{opt} \cup S^{\mathcal{G}}_{\overline{opt}}$ *where*
  - $S^{\mathcal{G}}_{opt} = S$*, and*
  - $S^{\mathcal{G}}_{\overline{opt}} = \{s^a \mid s \in S, a \in A(s)\}$*;*
- *for agent states $s \in S^{\mathcal{G}}_{opt}$, we have*
  - $A^{\mathcal{G}}(s) = A(s)$*,*
  - $\mathsf{P}^{\mathcal{G}}(s, a) = \{s^a \mapsto 1\}$ *for $a \in A^{\mathcal{G}}(s)$, i.e. it surely transitions to the newly added environment state, and*
  - $\mathsf{r}^{\mathcal{G}}(s, a) = \mathsf{r}(s, a)$ *for $a \in A^{\mathcal{G}}(s)$;*
- *for environment states $s^a \in S^{\mathcal{G}}_{\overline{opt}}$ we have*
  - $A^{\mathcal{G}}(s^a) = \mathcal{P}(s, a)$*, i.e. the uncertainty set,*
  - $\mathsf{P}^{\mathcal{G}}(s^a, \mathsf{P}) = \mathsf{P}$ *for $\mathsf{P} \in A^{\mathcal{G}}(s^a)$, and*
  - $\mathsf{r}^{\mathcal{G}}(s^a, \mathsf{P}) = \mathsf{r}_n(s, a)$ *for $\mathsf{P} \in A^{\mathcal{G}}(s^a)$, where r_n is an (objective-dependent)* neutral reward*.*

Intuitively, r_n is chosen in such a way that removing all $\overline{opt}$-states does not affect the Payoff of a path, i.e. for a an infinite path in the SG $\rho = s_0 a_0 s_1 a_1 \dots$ with $s_0 \in S^{\mathcal{G}}_{opt}$ we have $\mathsf{Payoff}(\rho) = \mathsf{Payoff}(s_0 a_0 s_2 a_2 \dots s_{2k} a_{2k} \dots)$. For TR objectives, the neutral reward is 0 and for LRA objectives, we define $\mathsf{r}_n(s^a) = r(s, a)$ for all $s^a \in S^{\mathcal{G}}_{\overline{opt}}$.

In general, this reduction results in an infinite-action SG, since $A^{\mathcal{G}}(s^a) = \mathcal{P}(s, a)$, and the uncertainty set commonly contains uncountably many distributions. However, for RMDPs with polytopic uncertainty sets, we can utilize the fact that the polytope can be captured by randomizing over its finitely many corner points. That is, each action inside the polytope can be simulated by a probabilistic policy randomly choosing between actions corresponding to corners of the polytope. This allows for a finite representation:

Definition 3 (Induced SG for polytopic RMDP)**.** *For a polytopic RMDP $\mathcal{M} = (S, A, \mathcal{P}, \mathsf{r})$ with $C(s, a) = \{\mathsf{P}^{s,a}_1, \dots, \mathsf{P}^{s,a}_k\} \subseteq \mathcal{P}(s, a)$ denoting the corner points of the polytopic confidence region for $\mathcal{P}(s, a)$, its induced SG $\mathcal{G}^{poly}_{\mathcal{M}} = (S^{\mathcal{G}}, A^{\mathcal{G}}, \mathsf{P}^{\mathcal{G}}, \mathsf{r}^{\mathcal{G}})$ can be obtained as in in Def. 2, only changing the available actions for environment states $s^a \in S^{\mathcal{G}}_{\overline{opt}}$ as $A^{\mathcal{G}}(s^a) = \{a^s_1, \dots, a^s_{|C(s,a)|}\}$, i.e. the corner points of the polytope.*

We prove the correctness of both reductions for all considered objectives, uncertainty sets, and semantics. Our proof is similar to (Chatterjee et al. 2024, Sec. 3.2); the novelty is the addition of TR objectives and the formalization of the infinite-action reduction, where the latter requires several changes.

Theorem 1 (Connection to SG – Proof in (Meggendorfer, Weininger, and Wienhöft 2024, App. C))**.** *Let $\mathcal{M}$ be an arbitrary RMDP and $\mathcal{G}_{\mathcal{M}}$ its induced* infinite-action *SG (Def. 2). Then for all $s \in S$ and any TR or LRA objective $\mathsf{V}^{opt}_{\mathcal{M}}(s) = \mathsf{V}^{opt}_{\mathcal{G}_{\mathcal{M}}}(s)$. Moreover, if $\mathcal{M}$ is polytopic and $\mathcal{G}'_{\mathcal{M}}$ its induced* finite-action *SG (Def. 3), $\mathsf{V}^{opt}_{\mathcal{G}_{\mathcal{M}}}(s) = \mathsf{V}^{opt}_{\mathcal{G}'_{\mathcal{M}}}(s)$.*

Implications of the Polytopic Reduction. Using this observation, we can immediately generalize (Chatterjee et al. 2024, Cor. 1) to undiscounted reward on polytopic RMDP: In finite-action SGs, memoryless deterministic policies are sufficient for optimizing TR objectives in SGs (Bertrand et al. 2023). With Thm. 1, we thus get that there is an optimal memoryless environment policy in RMDPs that is attained at the vertices of the polytope.

Corollary 1 (Environment Policy Semantics – Polytopic)**.** *In polytopic RMDPs with TR objectives, both agent and environment have deterministic memoryless optimal policies. Thus, stationary and time-varying semantics coincide.*

Complications for Arbitrary Uncertainty Sets. Finite-action SG have many useful properties, e.g. existence of memoryless deterministic optimal policies. For RMDPs with arbitrary uncertainty sets, this is not the case in general (see also (Grand-Clément, Petrik, and Vieille 2023, Prop. 3.2)):

Example 1 (Optimal Policy Need Not Exist)**.** *Consider the RMDP (in fact, a Robust Markov chain) in Fig. 1. The only action in state s_{init} has reward 0 and uncertainty set given by $0 \leq q = p^2$. The other states have values $\mathsf{V}(s_{goal}) = 1$ and $\mathsf{V}(s_{sink}) = 0$. Then the value of s_{init} is $V(s_{init}) = 0$ if $p = 0$ and $\frac{1}{1+p}$ otherwise. This function is discontinuous at $p = 0$. When the environment is maximizing (i.e. the agent is*

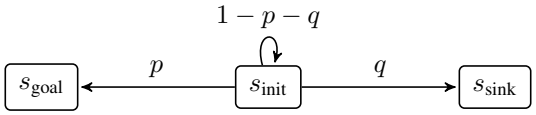

Figure 1: RMDP without optimal environment-policy.

minimizing costs), there is no optimal environment policy; the supremum over all environment policies is 1, but it cannot be attained. Even restricting to closed convex uncertainty sets is not sufficient: Intuitively, convex combinations can only increase q in relation to p, and thus decrease the value.

Sufficient Assumptions. So far, we have not put any restrictions on the uncertainty sets; Thm. 1 does not even require them to be closed or convex. However, for our approach we require that optimal policies exist for both agent and environment. To this end, we introduce a sufficient assumption, which will allow us to provide anytime algorithms with a stopping criterion.

***Constant-Support* Assumption:** *For all state-action pairs (s,a), two distributions in the same uncertainty set $\mathsf{P}_1, \mathsf{P}_2 \in \mathcal{P}(s,a)$ have the same support, i.e.* $\operatorname{supp}(\mathsf{P}_1) = \operatorname{supp}(\mathsf{P}_2)$.

Intuitively, this requires knowing all possible successors of each state-action pair, which is realistic in cases where the existence of transitions is certain and only their probability is unknown. In the context of robust systems, where typically it is known how the system behaves, this a natural assumption and is also called "positive uncertainty" (Chatterjee, Sen, and Henzinger 2008). In statistical applications, confidence intervals are derived by sampling an unknown system. Here, knowledge of the transition structure often is either assumed, also called "grey-box knowledge" (Ashok, Kretínský, and Weininger 2019), or can be ensured by gathering enough samples as many distance-based uncertainty sets, e.g. those in (Wang et al. 2024, Sec. 5.3), satisfy this assumption either naturally or for sufficiently small distances, see (Meggendorfer, Weininger, and Wienhöft 2024, App. A). A weaker assumption is to require only knowledge of the minimum positive transition probability (Daca et al. 2017). For this, we prove at the end of (Meggendorfer, Weininger, and Wienhöft 2024, App. C) how it reduces to the *Constant-Support* case and in the main body focus only on the latter for readability. We write *closed constant-support* RMDP to denote a closed RMDP satisfying the assumption. Note that polytopic RMDP and closed constant-support RMDP are incomparable: closed constant-support RMDP may be non-polytopic and polytopic RMDP may violate the *Constant-Support* Assumption.

Theorem 2 (Optimal Policies under *Constant-Support*)**.** *In every closed constant-support RMDP, optimal policies exist, formally* $\sup_\pi \inf_\tau \mathsf{V}^{\pi,\tau}_{\mathcal{M}}(s) = \max_\pi \min_\tau \mathsf{V}^{\pi,\tau}_{\mathcal{M}}(s)$*, and analogously for minimization objectives. Moreover, these policies are memoryless deterministic.*

Proof Sketch. Full Proof in (Meggendorfer, Weininger, and Wienhöft 2024, App. C). We first show value functions are continuous with respect to changes of the probabilities in the RMDP except for cases like Ex. 1 where the support of the transition changes the set of reachable states (Meggendorfer, Weininger, and Wienhöft 2024, Lem. 5). Intuitively, small changes to the probabilities can only have limited impact on the behaviour of an MDP, unless the change in probabilities adds or removes transitions, thereby potentially changing the fundamental long-term behaviour of the MDP, such as certain states being (un-)reachable from others. Thus, if the support for all distributions is constant, all value functions are continuous w.r.t the environment policy τ. Since all uncertainty sets (i.e. the sets of possible values of τ) are closed, the value function must admit optimal policies, as every continuous function attains its optimum on a closed domain. Then, existence of memoryless and deterministic optimal policies follow from basic observations on the objectives (Meggendorfer, Weininger, and Wienhöft 2024, Lem. 4). □

Corollary 2 (Environment Policy Semantics – Arbitrary)**.** *In closed constant-support RMDPs, stationary and time-varying semantics coincide.*

4 Implicit Bellman Updates

We now discuss how to solve RMDPs with value iteration (VI) in an *implicit* way. This means that we avoid constructing the induced SG explicitly, and the algorithm works directly on the RMDP. The motivation for this is twofold: Firstly, in polytopic RMDPs (given in $\mathcal{H}$-representation), the induced finite SG (and thus any approach solving it explicitly) requires exponential space. Secondly and more importantly, in general the induced SG has an *uncountably infinite* number of actions, and thus cannot be constructed explicitly at all.

Bellman Updates in SGs. VI centrally relies on the *Bellman update*. For example, for SGs with TR objective, we start from a lower bound L_0 on the value (e.g. $\mathsf{L}_0(s) = 0$ for all states s) and iteratively apply the update

$$\mathsf{L}_{i+1}(s) = opt_{a \in A(s)} \mathsf{r}(s,a) + \sum\nolimits_{s' \in S} \mathsf{P}(s,a)(s') \cdot \mathsf{L}_i(s'), \tag{1}$$

where $opt = \max$ if $s \in S_{max}$ and $opt = \min$ otherwise (Chen et al. 2013). Intuitively, this performs one step in the SG, back-propagating all rewards. In the limit, this sequence of estimates converges for TR objectives (Chen et al. 2013). Moreover, $\frac{1}{i}\mathsf{L}_i$ converges to the LRA value (Kretínský, Meggendorfer, and Weininger 2023b, Lem. 8).

Bellman Updates in RMDPs – Robust VI. Observe that in the induced SG, for any action that the agent chooses, the game surely transitions to the environment state corresponding to the chosen state-action pair, and it is the environment's turn to pick the uncertainty set. We can aggregate these two steps to one update in the RMDP by

$$\mathsf{L}_{i+1}(s) = opt_{a \in A(s)} \big(\mathsf{r}(s,a) + \overline{opt}_{\mathsf{P}(s,a) \in \mathcal{P}(s,a)} \sum\nolimits_{s' \in S} \mathsf{P}(s,a)(s') \cdot \mathsf{L}_i(s') \big) \tag{2}$$

Theorem 3 (Robust VI convergence – Proof in (Meggendorfer, Weininger, and Wienhöft 2024, App. D))**.** *Let $\mathcal{M}$ be a polytopic or closed constant-support RMDP. For a TR objective, the sequence L_i obtained from Eq.* (2) *converges*

to the value in the limit, i.e. for all $s \in S$ it holds that $\lim_{i\to\infty} \mathsf{L}_i(s) = \mathsf{V}^{opt}_{\mathcal{M}}(s)$. Similarly, for an LRA objective, $\lim_{i\to\infty} \frac{\mathsf{L}_i(s)}{i} = \mathsf{V}^{opt}_{\mathcal{M}}(s)$.

Eq. (2) generalizes robust VI for discounted reward as in, e.g. (Nilim and Ghaoui 2005). Unlike (Wang et al. 2024), we impose no restrictions on the structure of the RMDP. A result similar to this theorem is (Grand-Clément, Petrik, and Vieille 2023, Thm. 5.2), which shows convergence for RMDPs with "definable" uncertainty and LRA objective.

Implicit Updates. By itself, Eq. (2) is only of theoretical value for now, as $\mathcal{P}(s,a)$ might be uncountably infinite. The key to an effective algorithm is the ability to evaluate the inner expression in Eq. (2). This only requires optimizing a linear function over the uncertainty set (generalizing the already quite generic "definable" assumption (Grand-Clément, Petrik, and Vieille 2023, Def. 4.11)). Moreover, this optimization can be performed efficiently for many uncertainty sets, of which we provide a (non-exhaustive) list.

Lemma 1 (Efficiency of the Implicit Update)**.** *Let $\mathcal{P}(s,a)$ be an uncertainty set given as (i) polytope in $\mathcal{H}$-representation or (ii) $\mathcal{V}$-representation, or (iii) (weighted) L^p-norm-balls around a probability distribution where $p \in \mathbb{N} \cup \{\infty\}$. Then,*

$$\overline{opt}_{\mathsf{P}(s,a)\in\mathcal{P}(s,a)} \sum\nolimits_{s'\in S} \mathsf{P}(s,a)(s') \cdot \mathsf{L}_i(s')$$

can be evaluated with a number of operations that is polynomial w.r.t. $|S|$ and the representation of $\mathcal{P}(s,a)$.

Proof Sketch. Full Proof in (Meggendorfer, Weininger, and Wienhöft 2024, App. D). Case (i) reduces to a polynomially sized linear program, which is PTIME (Karmarkar 1984). For (ii), iterating over all vertices takes linear time. For L^1- and L^∞-balls it suffices to order the successors according to L_i and maximize their probability in this order. For general L^p-balls we compute the surface point where the gradient of the objective function is orthogonal to its surface. We note that when considering interval constraints (a special case of (iii)), this coincides with the technique of *ordering-maximization* (Givan, Leach, and Dean 2000; Lahijanian, Andersson, and Belta 2015), which is also employed by `IntervalMDP.jl` (Mathiesen, Lahijanian, and Laurenti 2024). □

Thus, for a large class of RMDPs (with any of the listed uncertainty sets and LRA or TR objectives), every single step of VI is fast. Further, while the overall number of steps for VI can be exponential, typically a much smaller number of iterations suffices (Hartmanns et al. 2023), suggesting that implicit Bellman updates yields an efficient VI approach. Together, this gives rise to a generic implicit value iteration algorithm, presented in Alg. 1. Thm. 3 directly yields that the computed L_i converge to the true value in the limit and Lem. 1 shows that the updates in Line 4 can be performed effectively and efficiently. However, note that such a (one-sided) VI does not yet give us a stopping criterion. In particular, while the rule in Line 5 usually works well in practice, it does not guarantee that the computed values are close to the true value. Obtaining such a guarantee is the topic of the next section.

Algorithm 1 Best-Effort Implicit Value Iteration for RMDP

Input: Polytopic or closed constant-support RMDP $\mathcal{M}$ and precision hint $\varepsilon > 0$

Output: Lower bounds on the optimal total reward value $\mathsf{V}^{opt}_{\mathcal{M}}$

1: $\mathsf{L}_0(\cdot) \leftarrow 0$, $i \leftarrow 0$
2: **while** `true` **do**
3: **for all** $s \in S$ **do**
4: $\mathsf{L}_{i+1}(s) \leftarrow opt_{a\in A(s)} \big(\mathsf{r}(s,a) + \overline{opt}_{\mathsf{P}(s,a)\in\mathcal{P}(s,a)} \sum_{s'\in S} \mathsf{P}(s,a)(s') \cdot \mathsf{L}_i(s')\big)$
5: **if** $\max_{s\in S} \mathsf{L}_{i+1}(s) - \mathsf{L}_i(s) < \varepsilon$ **then**
6: **return** L_{i+1}
7: $i \leftarrow i+1$

5 Implicit Anytime Value Iteration

The approach of Sec. 4 converges in the limit (similar to (Grand-Clément, Petrik, and Vieille 2023; Wang et al. 2024)), however we cannot bound the distance between L_i and $\mathsf{V}^{opt}_{\mathcal{M}}$ for any concrete i. In other words, we do not know how close we are to the true value at any time and thus cannot give any guarantees upon stopping the VI. This absence of a stopping criterion is explicitly noted as an open question in (Grand-Clément, Petrik, and Vieille 2023, Sec. 5.2). Even for non-robust systems, efficient stopping criteria were a major challenge. One prominent solution is *bounded value iteration (BVI)*, see e.g. (Kretínský, Meggendorfer, and Weininger 2023a). The main idea is to compute an additional sequence of *upper* bounds U_i that *over*-approximates the value and converges to it in the limit, yielding an *anytime algorithm*. Our goal is to obtain such an algorithm for RMDP.

Definition 4 (Anytime Algorithm with Stopping Criterion)**.** *An anytime algorithm (with stopping criterion) for RMDPs maintains two sequences $\mathsf{L}_i, \mathsf{U}_i$ such that for all states s (i) for every iteration $i \in \mathbb{N}$, $\mathsf{L}_i(s) \leq \mathsf{V}^{opt}_{\mathcal{M}}(s) \leq \mathsf{U}_i(s)$, and (ii) $\lim_{i\to\infty} \mathsf{U}_i(s) - \mathsf{L}_i(s) = 0$.*

Intuitively, an anytime algorithm is correct at every step and guarantees a precision of $\mathsf{U}_i(s) - \mathsf{L}_i(s)$; moreover, eventually the algorithm terminates for every precision $\varepsilon > 0$. Using Thm. 1, we can obtain an *explicit* anytime algorithm for polytopic RMDPs, namely by constructing the induced finite-action SG and applying the algorithms of (Kretínský, Meggendorfer, and Weininger 2023a).

Key Contribution. To obtain an algorithm that is efficient and applicable for arbitrary uncertainty sets, we now propose an *implicit* anytime algorithm. For ease of presentation, the descriptions in this section focus on closed constant-support RMDPs with TR objective. We later provide an intuition how to extend the results to LRA objectives and non-constant support RMDPs (details are provided in (Meggendorfer, Weininger, and Wienhöft 2024, Apps. E and F)).

Challenges. Obtaining converging upper bounds is not as simple as for lower bounds. In particular, just applying Bellman updates to upper bounds does not necessarily converge.

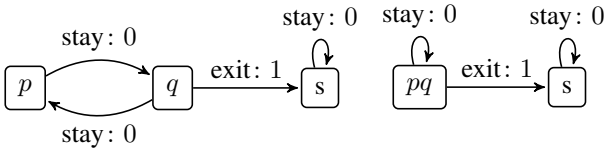

Figure 2: An MDP where value iteration from above does not converge (left) and the collapsed MDP (right).

Example 2 (Non-Convergence of Upper Bounds)**.** *Consider the RMDP (even MDP) in Fig. 2 (left) and assume BVI starts with an upper bound of* $\mathsf{U}_0(p) = \mathsf{U}_0(q) = t > 1$*. The correct value is* $\mathsf{V}^{\max}_{\mathcal{M}} = 1$*, since* p *has the value of* q*, and* q *can pick action* $exit$ *to obtain a reward of 1 and enter the sink* s*, from which point onward no further rewards are collected. However, the Bellman update in* q *chooses the action* $stay$ *that maximizes the upper bound, keeping it at* t*. Thus,* $\mathsf{U}(p) = \mathsf{U}(q) = t$ *is a spurious fixpoint of the Bellman updates for* all $t > 1$*, and BVI does not converge from above.*

Convergence by Collapsing End Components. The core problem are so-called *end components (ECs)*, e.g. (De Alfaro 1997, Chp. 3.3), which are cyclic parts of the state space where the agent can remain arbitrarily long without obtaining any reward. The solution introduced in (Brázdil et al. 2014; Haddad and Monmege 2018) is to *collapse* these ECs, i.e. to replace them with a single representative and remove all internal behaviour. In the example, the states p and q can be aggregated into a single state while removing the $stay$ actions, as depicted in Fig. 2 (right). The modified MDP has the same value (where all collapsed states have the value of their representative) and in it, Bellman updates have a unique fixpoint so that BVI converges. Since in constant-support RMDPs the environment cannot affect the set of successor states, ECs are solely under the agents control and we can lift the solution of collapsing from MDP to RMDP.

Lemma 2 (Collapsing – Proof in (Meggendorfer, Weininger, and Wienhöft 2024, App. E))**.** *Let* $\mathcal{M}$ *be a closed constant-support RMDP with a TR objective. We can construct a linearly sized RMDP* $\mathcal{M}' = \mathsf{COLLAPSE}(\mathcal{M})$ *in polynomial time, such that* $\mathsf{V}^{opt}_{\mathcal{M}} = \mathsf{V}^{opt}_{\mathcal{M}'}$ *and in* $\mathcal{M}'$*, the Bellman updates have a unique fixpoint.*

Initializing Upper Bounds. Classical BVI requires an (a-priori) upper bound U that for all states over-approximates their value. Such a bound can be obtained in two steps, which we only briefly outline in the interest of space. First, we identify states with infinite value, which can be done by graph analysis, extending methods for SG (Chen et al. 2013) to work *implicitly* in RMDPs. The remaining states with finite value almost surely reach a sink state where no further reward is obtained. Their value can be bounded by extending a standard approach to our setting, see e.g. (Kretínský, Meggendorfer, and Weininger 2023b, App. B). In essence, in closed constant-support RMDPs we have a positive lower bound on all transition probabilities with which we can conservatively bound the expected number of steps until a sink state is reached and assume that until then the maximal single-step reward is obtained. We call this procedure $\mathsf{INIT\_TR}$ and formally describe it in (Meggendorfer, Weininger, and Wienhöft 2024, App. E).

Algorithm 2 Bounded Value Iteration for RMDP

Input: closed constant-support RMDP $\mathcal{M}$ and desired precision $\varepsilon > 0$

Output: ε-precise lower and upper bounds L and U on the optimal total reward value $\mathsf{V}^{opt}_{\mathcal{M}}$

$\mathcal{M}' \leftarrow \mathsf{COLLAPSE}(\mathcal{M})$
$\mathsf{L}_0, \mathsf{U}_0 \leftarrow \mathsf{INIT\_TR}(\mathcal{M}')$
$i \leftarrow 0$
while $\mathsf{U}(s_i) - \mathsf{L}(s_i) > \varepsilon$ **do**
  for all $s \in S'$ **do**
    $\mathsf{L}_{i+1}(s) \leftarrow opt_{a \in A(s)} \mathsf{r}(s,a) + \overline{opt}_{\mathsf{P}(s,a) \in \mathcal{P}(s,a)} \sum_{s' \in S} \mathsf{P}(s,a)(s') \cdot \mathsf{L}_i(s')$
    $\mathsf{U}_{i+1}(s) \leftarrow opt_{a \in A(s)} \mathsf{r}(s,a) + \overline{opt}_{\mathsf{P}(s,a) \in \mathcal{P}(s,a)} \sum_{s' \in S} \mathsf{P}(s,a)(s') \cdot \mathsf{U}_i(s')$
  $i \leftarrow i + 1$
return $(\mathsf{L}_i, \mathsf{U}_i)$

Lemma 3 ($\mathsf{INIT\_TR}$ – Proof in (Meggendorfer, Weininger, and Wienhöft 2024, App. E))**.** *Let* $\mathcal{M}$ *be a polytopic or closed constant-support RMDP. There exists a procedure* $\mathsf{INIT\_TR}$ *that for* $\mathcal{M}$ *and a TR objective computes functions* L, U *such that (i) all states* s *with infinite value have* $\mathsf{L}(s) = \mathsf{U}(s) = \infty$ *and (ii) all states* s *with finite value have* $\mathsf{L}(s) = 0$ *and* $\mathsf{U}(s) = t$*, where* $t \in \mathbb{Q}$ *is an upper bound on the maximum finite expected total reward.*

In practice, *optimistic value iteration* (OVI) (Hartmanns and Kaminski 2020; Azeem et al. 2022) has proven to be more efficient. Instead of fixing upper bounds a-priori, we adaptively guess and verify them, using effectively the same building blocks. The details of OVI are rather involved and we avoid discussion to not distract from the key results. For scalability, our implementation uses this approach.

Algorithm. Alg. 2 shows the overall BVI algorithm for TR objectives. First, it collapses ECs to ensure that Bellman updates have a unique fixpoint and computes the initial bounds. Then the main loop applies implicit Bellman updates Eq. (2). Since by Lem. 3 the initial bounds are correct and by Lem. 2 Bellman updates have a unique fixpoint, the lower and upper bounds eventually become arbitrarily close.

Long-Run Average Reward. For closed constant-support RMDPs with LRA objectives, a very similar construction is possible based on (Ashok et al. 2017). We modify the $\mathsf{COLLAPSE}$ procedure as follows (see (Meggendorfer, Weininger, and Wienhöft 2024, App. E) for the formal description): When replacing an EC, add an action to its representative that leads to a sink state and as reward obtains the value of staying in the EC forever. Thus, playing this action in the modified RMDP corresponds to playing optimally in the EC of the original RMDP, and thus preserves the values. In this way, we can reduce LRA objectives to TR objectives and then apply Alg. 2.

Theorem 4 (Implicit Anytime Algorithm with Stopping Criterion – Proof in (Meggendorfer, Weininger, and Wienhöft 2024, App. E))**.** *For every closed constant-support RMDP $\mathcal{M}$ with TR objective and precision $\varepsilon > 0$, Alg. 2 is an anytime algorithm (Def. 4). For LRA objectives, its modification (see (Meggendorfer, Weininger, and Wienhöft 2024, App. E)) is an anytime algorithm. Both algorithms work* implicitly*, i.e. without constructing the induced SG.*

Beyond *Constant-Support*. In RMDPs where the environment can affect the successors of an action, the solution of collapsing is not applicable anymore. In particular, states in an EC may have different values, as the environment can "lock" the agent in subpart of the EC. This was the key complication for developing stopping criteria for SGs, see e.g. (Kretínský, Meggendorfer, and Weininger 2023a, Sec. III-B), necessitating additional analysis of the ECs. Moreover, we aim to do so implicitly, adding another layer of complexity. In (Meggendorfer, Weininger, and Wienhöft 2024, App. F) we provide implicit anytime algorithms for SSP and maximizing TR and explain the complications for LRA and minimizing TR, for which we may need to resort to an exponential blowup by enumerating possible supports. We conjecture that this can be avoided but leave this question for future work.

Optimal Policies. Often, we are not only interested in the value of an RMDP, but also optimal policies. These can be derived without computational overhead for both the agent and the environment: Using the connection between RMDP and SG, we can apply (Kretínský, Meggendorfer, and Weininger 2023a, Lem. 1). Intuitively, during every Bellman update, we remember the optimal action for every state. Then, when the algorithm terminates, these actions form an optimal policy.

6 Experimental Evaluation

We implemented a prototype in Java, based on `PET` (Meggendorfer and Weininger 2024). For linear optimization, we use the (pure Java) library `oj! Algorithms`. We ran our experiments on a machine with standard hardware (AMD Ryzen 5 CPU, 16GB RAM) running Linux Mint OS and using OpenJDK 21 as JRE. Our tool, its source code, all models, and instructions to replicate all results can be found at (Meggendorfer 2024).

Features. For uncertainty sets, our prototype supports (i) linear constraints, i.e. $\mathcal{H}$-representation of polytopes, (ii) rectangular constraints, i.e. lower and upper bounds for probabilities, and (iii) norm-based constraints, i.e. giving a centre point q together with a radius r and including all probabilities with L^1, L^2, or L^∞ distances at most r from q. (Note that both L^1 and L^∞ constraints are a special case of linear constraints, but L^1 constraints can be exponentially more succinct.)

In terms of models, our tool supports three formats. Firstly, a simple, explicit format in JSON (described in the artefact). Secondly, we can consider a *robust* variant of `PRISM` models (Kwiatkowska, Norman, and Parker 2011), obtained by adding L^1, L^2, or L^∞ balls of a given radius on each action. Finally, our tool supports interval models in `PRISM` language (i.e. rectangular constraints).

For properties, our tool supports maximizing and minimizing LRA, TR, and SSP. For LRA and minimizing TR, *Constant-Support* is required (due to the discussion above).

Models. We consider several sources of models. Firstly, we handcrafted ten models where we could manually derive the correct value to validate our tool, in particular covering many corner-cases. Since these models are small, we do not include them for performance evaluation. Secondly, we use models from (Chatterjee et al. 2024) and their scaled up versions. Finally, we modified several standard models from (Hartmanns et al. 2019) by adding rectangular constraints on selected transitions or adding L^p balls around all transitions. All considered (and further) models are included in the artefact.

Previous Approaches. For (Grand-Clément, Petrik, and Vieille 2023; Wang et al. 2024), neither code nor case studies are available online. We consider the implementation of (Chatterjee et al. 2024), denoted `RPPI`, which also implements the approaches of (Wang et al. 2023), called `RVI` and `RRVI`. We mention a few caveats: Our tool is implemented in Java, while the implementation of (Chatterjee et al. 2024) is written in Python and uses `stormpy` (Python binding for the model checker `Storm` (Hensel et al. 2022)), which might be a source of performance differences. Moreover, our input formats are fundamentally different: The approach of (Chatterjee et al. 2024) only supports linear constraints and assumes that the vertices of the constraint set is explicitly given, i.e. in $\mathcal{V}$-representation. In contrast, we support many different representations. Naturally, the performance of an algorithm depends on the appropriate input representation, and one could argue that choosing different input formats is giving an unfair advantage. However, constraint sets given by L^p-balls or rectangular constraints are the typical use-case for RMDP. In particular, the implementation of (Chatterjee et al. 2024) explicitly generates the $\mathcal{V}$-representation from L^1-balls or rectangular constraints. As such, we represent our model instances in this implicit way.

As a further competitor, `PRISM` (Kwiatkowska, Norman, and Parker 2011), a state-of-the-art probabilistic model checker, supports TR objectives. They also use a value iteration based approach, however it does not provide guarantees and only works on models with rectangular constraints and *Constant-Support*. On the considered models our results coincided with those of `PRISM`, giving further indication for the correctness of our approach and implementation.

Guarantees and Runtime. We remark that `RVI` and `RRVI` as well as `PRISM` do not give a practical stopping criterion. For the former two, the implementation from (Chatterjee et al. 2024) aids them by stopping once the iterates are sufficiently close to the correct value, while `PRISM` stops once the iterates do not changes much between steps. Notably, `PRISM`'s heuristic can indeed lead to stopping early and wrongly, see (Haddad and Monmege 2018). In contrast, our approach produces converging lower and upper bounds and thus also provides a correct stopping criterion. Computing both bounds until achieving precision of ε (10^{-6} in our experiments) naturally requires more effort than just working with one side and stopping at the correct time via an oracle or heuristics:

Model	Ours	RVI	RRVI	RPPI
cont-50	<1	223	<1	30
cont-75	<1	700	<1	70
cont-100	<1	M/O		
cont-125	1	M/O		
lake-10-U	<1	–	T/O	7
lake-10-M	<1	–	–	26
lake-15-U	<1	–	T/O	136
lake-15-M	2	–	–	T/O
lake-100-U	6	–	T/O	T/O
lake-100-M	12	–	–	T/O

Table 1: Comparison of our approach for maximizing LRA to the approaches implemented in (Chatterjee et al. 2024) on the models used in that paper. We report solving times (excluding model building / parsing) in seconds. A dash indicates the approach does not support a particular model. T/O denotes a timeout of 15 minutes, M/O a memory-out crash. lake-n are their **frozenlake** models of size $n \times n$, the suffix U or M indicates the unichain or multichain variant. cont-n are the **contamination** models with n states.

Firstly, BVI needs to perform twice as many operations in each step (updating lower *and* upper bound). Secondly, the lower bound may already (unknowingly) have converged to the correct value (leading to PRISM stopping), yet the upper bound may still require further updates until convergence. Intuitively, PRISM's approach "only" proves that a certain lower bound is achievable, while BVI additionally proves that nothing better is possible.

Experimental Results

Table 1 shows that our approach massively outperforms RVI, RRVI, and RRPI by several orders of magnitude. In particular, it seems that the runtime of their approaches grow significantly faster than ours. For example, going from lake-10-M to lake-15-M increases the runtime of RPPI by a factor of about 50, while ours only increases by a factor of ≈5. We believe this is partly due to implementation inefficiencies, but more importantly due the exponential space requirement of constructing the induced SG explicitly. In particular, just building the game structure for lake-100-U in their implementation takes 200s, and crashes with a memory-out on cont-100.

In Table 2, we compare to the approach of PRISM. Our approach has runtimes in the same order of magnitude, with differences likely due to implementation details. However, recall that *by design* our tool needs to work more, as it provides guarantees. Thus, these results demonstrate that additionally obtaining guarantees via our approach does not drastically increase the runtime and scales to significantly sized models.

Finally, we also evaluated our tool on robust variants of large, established models by adding L^1- and L^2-norm balls. As there are no competing tools, we delegate details to (Meggendorfer, Weininger, and Wienhöft 2024, App. G) in the interest of space. In brief, we observed that our approach

Model	$\lvert S\rvert$	$\lvert A\rvert$	Ours	PRISM
firewire	46,878	92,144	78	111
frozenlake	22,500	90,000	52	25
lake_swarm	20,736	82,944	60	68
brp	217,155	217,155	88	40

Table 2: Comparison of our approach to PRISM. Columns $|S|$ and $|A|$ denote the number of states and actions in the model, respectively. We report solving times in seconds as in Table 1. The models are obtained by adding rectangular constraints to existing ones. Moreover, all experiments use a TR objective (as PRISM only supports these for RMDP). Note that our tool gives guarantees and PRISM does not, thus higher runtimes are to be expected (see previous discussion).

can efficiently handle complicated uncertainty sets such as L^2-balls – out of reach for state-of-the-art tools, and it can solve models with over a million states in under a minute.

IntervalMDP.jl. Finally, for completeness we also tried evaluating IntervalMDP.jl (Mathiesen, Lahijanian, and Laurenti 2024) on the models considered in Table 2. However, the tool ran out of memory for each model already when loading the model (even before specifying an objective). We conjecture that this is due to the tool working with the (large) explicit representation and not the PRISM language directly. However, we again emphasize that their focus is quite different, and as such we cannot draw meaningful conclusions.

7 Conclusion

We have generalized the connection between RMDPs and SGs to include arbitrary uncertainty sets and total reward objectives, we have shown that and how Bellman updates can be performed implicitly and efficiently, and we have provided anytime algorithms with stopping criteria. Together, we have presented a framework for solving RMDPs that is generic, reliable and efficient. In the future, we aim to to investigate what form solutions can take when lifting the *Constant-Support* Assumption.

Acknowledgements

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 101034413, the ERC CoG 863818 (ForM-SMArt), and the DFG through the Cluster of Excellence EXC 2050/1 (CeTI, project ID 390696704, as part of Germany's Excellence Strategy) and the TRR 248 (see https://perspicuous-computing.science, project ID 389792660).

References

Ashok, P.; Chatterjee, K.; Daca, P.; Kretínský, J.; and Meggendorfer, T. 2017. Value Iteration for Long-Run Average Reward in Markov Decision Processes. In *CAV (1)*, volume 10426 of *Lecture Notes in Computer Science*, 201–221. Springer.

Ashok, P.; Kretínský, J.; and Weininger, M. 2019. PAC Statistical Model Checking for Markov Decision Processes and Stochastic Games. In *CAV, Part I*, volume 11561 of *LNCS*, 497–519. Springer.

Azeem, M.; Evangelidis, A.; Kretínský, J.; Slivinskiy, A.; and Weininger, M. 2022. Optimistic and Topological Value Iteration for Simple Stochastic Games. In *ATVA*, volume 13505 of *Lecture Notes in Computer Science*, 285–302. Springer.

Badings, T. S.; Simão, T. D.; Suilen, M.; and Jansen, N. 2023. Decision-making under uncertainty: beyond probabilities. *Int. J. Softw. Tools Technol. Transf.*, 25(3): 375–391.

Baier, C.; Klein, J.; Leuschner, L.; Parker, D.; and Wunderlich, S. 2017. Ensuring the Reliability of Your Model Checker: Interval Iteration for Markov Decision Processes. In *CAV (1)*, volume 10426 of *Lecture Notes in Computer Science*, 160–180. Springer.

Bertrand, N.; Bouyer-Decitre, P.; Fijalkow, N.; and Skomra, M. 2023. Stochastic Games. In Fijalkow, N., ed., *Games on Graphs*.

Brázdil, T.; Chatterjee, K.; Chmelik, M.; Forejt, V.; Křetínský, J.; Kwiatkowska, M. Z.; Parker, D.; and Ujma, M. 2014. Verification of Markov Decision Processes Using Learning Algorithms. In *ATVA*, 98–114. Springer.

Buffet, O. 2005. Reachability Analysis for Uncertain SSPs. In *ICTAI*, 515–522. IEEE Computer Society.

Chatterjee, K.; Goharshady, E. K.; Karrabi, M.; Novotný, P.; and Zikelic, D. 2023. Solving Long-run Average Reward Robust MDPs via Stochastic Games. *CoRR*, abs/2312.13912.

Chatterjee, K.; Goharshady, E. K.; Karrabi, M.; Novotný, P.; and Zikelic, D. 2024. Solving Long-run Average Reward Robust MDPs via Stochastic Games. In Larson, K., ed., *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, 6707–6715. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Chatterjee, K.; Sen, K.; and Henzinger, T. A. 2008. Model-Checking omega-Regular Properties of Interval Markov Chains. In *FoSSaCS*, volume 4962 of *Lecture Notes in Computer Science*, 302–317. Springer.

Chen, T.; Forejt, V.; Kwiatkowska, M. Z.; Parker, D.; and Simaitis, A. 2013. Automatic verification of competitive stochastic systems. *Formal Methods Syst. Des.*, 43(1): 61–92.

Condon, A. 1992. The Complexity of Stochastic Games. *Inf. Comput.*, 96(2): 203–224.

Daca, P.; Henzinger, T. A.; Kretínský, J.; and Petrov, T. 2017. Faster Statistical Model Checking for Unbounded Temporal Properties. *ACM Trans. Comput. Log.*, 18(2): 12:1–12:25.

De Alfaro, L. 1997. *Formal verification of probabilistic systems*. Ph.D. thesis, Stanford university.

Givan, R.; Leach, S.; and Dean, T. 2000. Bounded-parameter Markov decision processes. *Artificial Intelligence*, 122(1-2): 71–109.

Grand-Clément, J.; Petrik, M.; and Vieille, N. 2023. Beyond discounted returns: Robust Markov decision processes with average and Blackwell optimality. *CoRR*, abs/2312.03618.

Grünbaum, B. 2003. *Convex Polytopes*. Springer, second edition. ISBN 978-0-387-00424-2.

Haddad, S.; and Monmege, B. 2018. Interval iteration algorithm for MDPs and IMDPs. *Theor. Comput. Sci.*, 735: 111–131.

Hartmanns, A.; Junges, S.; Quatmann, T.; and Weininger, M. 2023. A Practitioner's Guide to MDP Model Checking Algorithms. In *TACAS (1)*, volume 13993 of *Lecture Notes in Computer Science*, 469–488. Springer.

Hartmanns, A.; and Kaminski, B. L. 2020. Optimistic Value Iteration. In *CAV (2)*, volume 12225 of *Lecture Notes in Computer Science*, 488–511. Springer.

Hartmanns, A.; Klauck, M.; Parker, D.; Quatmann, T.; and Ruijters, E. 2019. The Quantitative Verification Benchmark Set. In *TACAS (1)*, volume 11427 of *Lecture Notes in Computer Science*, 344–350. Springer.

Hensel, C.; Junges, S.; Katoen, J.; Quatmann, T.; and Volk, M. 2022. The probabilistic model checker Storm. *Int. J. Softw. Tools Technol. Transf.*, 24(4): 589–610.

Ho, C. P.; Petrik, M.; and Wiesemann, W. 2018. Fast Bellman Updates for Robust MDPs. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, 1984–1993. PMLR.

Iyengar, G. N. 2005. Robust Dynamic Programming. *Math. Oper. Res.*, 30(2): 257–280.

Karmarkar, N. 1984. A new polynomial-time algorithm for linear programming. *Comb.*, 4(4): 373–396.

Kretínský, J.; Meggendorfer, T.; and Weininger, M. 2023a. Stopping Criteria for Value Iteration on Stochastic Games with Quantitative Objectives. In *LICS*, 1–14. IEEE.

Kretínský, J.; Meggendorfer, T.; and Weininger, M. 2023b. Stopping Criteria for Value Iteration on Stochastic Games with Quantitative Objectives. *CoRR*, abs/2304.09930.

Kwiatkowska, M.; Norman, G.; and Parker, D. 2011. PRISM 4.0: Verification of Probabilistic Real-time Systems. In Gopalakrishnan, G.; and Qadeer, S., eds., *Proc. 23rd International Conference on Computer Aided Verification (CAV'11)*, volume 6806 of *LNCS*, 585–591. Springer.

Lahijanian, M.; Andersson, S. B.; and Belta, C. 2015. Formal Verification and Synthesis for Discrete-Time Stochastic Systems. *IEEE Trans. Autom. Control.*, 60(8): 2031–2045.

Mathiesen, F. B.; Lahijanian, M.; and Laurenti, L. 2024. IntervalMDP.jl: Accelerated Value Iteration for Interval Markov Decision Processes. *IFAC-PapersOnLine*, 58(11): 1–6. ADHS 2024.

Meggendorfer, T. 2024. Solving Robust Markov Decision Processes: Generic, Reliable, Efficient (artefact). Zenodo. https://doi.org/10.5281/zenodo.14385450.

Meggendorfer, T.; and Weininger, M. 2024. Playing Games with Your PET: Extending the Partial Exploration Tool to Stochastic Games. In *CAV (3)*, volume 14683 of *Lecture Notes in Computer Science*, 359–372. Springer.

Meggendorfer, T.; Weininger, M.; and Wienhöft, P. 2024. Solving Robust Markov Decision Processes: Generic, Reliable, Efficient. *CoRR*, abs/2412.10185.

Nilim, A.; and Ghaoui, L. E. 2005. Robust Control of Markov Decision Processes with Uncertain Transition Matrices. *Oper. Res.*, 53(5): 780–798.

Puterman, M. L. 1994. *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley and Sons.

Sen, K.; Viswanathan, M.; and Agha, G. 2006. Model-Checking Markov Chains in the Presence of Uncertainties. In *TACAS*, volume 3920 of *Lecture Notes in Computer Science*, 394–410. Springer.

Strehl, A. L.; and Littman, M. L. 2004. An empirical evaluation of interval estimation for Markov decision processes. In *16th IEEE International Conference on Tools with Artificial Intelligence*, 128–135. IEEE.

Tewari, A.; and Bartlett, P. L. 2007. Bounded Parameter Markov Decision Processes with Average Reward Criterion. In *COLT*, volume 4539 of *Lecture Notes in Computer Science*, 263–277. Springer.

Wang, Y.; Velasquez, A.; Atia, G. K.; Prater-Bennette, A.; and Zou, S. 2023. Robust Average-Reward Markov Decision Processes. In Williams, B.; Chen, Y.; and Neville, J., eds., *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, 15215–15223. AAAI Press.

Wang, Y.; Velasquez, A.; Atia, G. K.; Prater-Bennette, A.; and Zou, S. 2024. Robust Average-Reward Reinforcement Learning. *J. Artif. Intell. Res.*, 80: 719–803.

Wu, D.; and Koutsoukos, X. D. 2008. Reachability analysis of uncertain systems using bounded-parameter Markov decision processes. *Artif. Intell.*, 172(8-9): 945–954.

Yang, I. 2017. A Convex Optimization Approach to Distributionally Robust Markov Decision Processes With Wasserstein Distance. *IEEE Control Systems Letters*, 1(1): 164–169.

Zhang, Y.; Steimle, L.; and Denton, B. T. 2017. Robust Markov decision processes for medical treatment decisions. *Optimization online*.