Sound Value Iteration

Tim Quatmann(✉) and Joost-Pieter Katoen

RWTH Aachen University, Aachen, Germany
tim.quatmann@cs.rwth-aachen.de

Abstract. Computing reachability probabilities is at the heart of probabilistic model checking. All model checkers compute these probabilities in an iterative fashion using value iteration. This technique approximates a fixed point from below by determining reachability probabilities for an increasing number of steps. To avoid results that are significantly off, variants have recently been proposed that converge from both below and above. These procedures require starting values for both sides. We present an alternative that does not require the a priori computation of starting vectors and that converges faster on many benchmarks. The crux of our technique is to give tight and safe bounds—whose computation is cheap—on the reachability probabilities. Lifting this technique to expected rewards is trivial for both Markov chains and MDPs. Experimental results on a large set of benchmarks show its scalability and efficiency.

1 Introduction

Markov decision processes (MDPs) [1,2] have their roots in operations research and stochastic control theory. They are frequently used for stochastic and dynamic optimization problems and are widely applicable in, e.g., stochastic scheduling and robotics. MDPs are also a natural model in randomized distributed computing where coin flips by the individual processes are mixed with non-determinism arising from interleaving the processes' behaviors. The central problem for MDPs is to find a policy that determines what action to take in the light of what is known about the system at the time of choice. The typical aim is to optimize a given objective, such as minimizing the expected cost until a given number of repairs, maximizing the probability of being operational for 1,000 steps, or minimizing the probability to reach a "bad" state.

Probabilistic model checking [3,4] provides a scalable alternative to tackle these MDP problems, see the recent surveys [5,6]. The central computational issue in MDP model checking is to solve a system of linear inequalities. In absence of non-determinism—the MDP being a Markov Chain (MC)—a linear equation system is obtained. After appropriate pre-computations, such as determining the states for which no policy exists that eventually reaches the goal state, the (in)equation system has a unique solution that coincides with the extremal value

This work is partially supported by the Sino-German Center project CAP (GZ 1023).

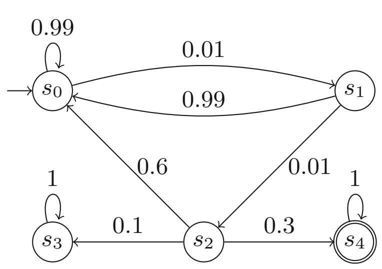
that is sought for. Possible solution techniques to compute such solutions include policy iteration, linear programming, and value iteration. Modern probabilistic model checkers such as `PRISM` [7] and `Storm` [8] use value iteration by default. This approximates a fixed point from below by determining the probabilities to reach a target state within k steps in the k-th iteration. The iteration is typically stopped if the difference between the value vectors of two successive (or vectors that are further apart) is below the desired accuracy ε.

This procedure however can provide results that are significantly off, as the iteration is stopped prematurely, e.g., since the probability mass in the MDP only changes slightly in a series of computational steps due to a "slow" movement. This problem is not new; similar problems, e.g., occur in iterative approaches to compute long-run averages [9] and transient measures [10] and pop up in statistical model checking to decide when to stop simulating for unbounded reachability properties [11]. As recently was shown, this phenomenon does not only occur for hypothetical cases but affects practical benchmarks of MDP model checking too [12]. To remedy this, Haddad and Monmege [13] proposed to iteratively approximate the (unique) fixed point from both below and above; a natural termination criterion is to halt the computation once the two approximations differ less than $2{\cdot}\varepsilon$. This scheme requires two starting vectors, one for each approximation. For reachability probabilities, the conservative values zero and one can be used. For expected rewards, it is non-trivial to find an appropriate upper bound—how to "guess" an adequate upper bound to the expected reward to reach a goal state? Baier *et al.* [12] recently provided an algorithm to solve this issue.

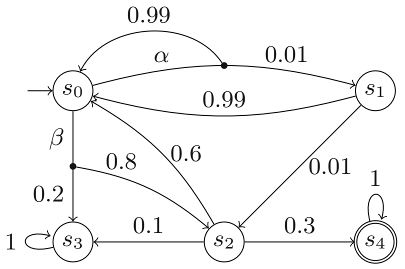
This paper takes an alternative perspective to obtaining a sound variant of value iteration. *Our approach does not require the a priori computation of starting vectors and converges faster on many benchmarks.* The crux of our technique is to give tight and safe bounds—whose computation is cheap and that are obtained during the course of value iteration—on the reachability probabilities. The approach is simple and can be lifted straightforwardly to expected rewards. The central idea is to split the desired probability for reaching a target state into the sum of

(i) the probability for reaching a target state *within* k steps and
(ii) the probability for reaching a target state *only after* k steps.

We obtain (i) via k iterations of (standard) value iteration. A second instance of value iteration computes the probability that a target state is still reachable after k steps. We show that from this information safe lower and upper bounds for (ii) can be derived. We illustrate that the same idea can be applied to expected rewards, topological value iteration [14], and Gauss-Seidel value iteration. We also discuss in detail its extension to MDPs and provide extensive experimental evaluation using our implementation in the model checker `Storm` [8]. Our experiments show that on many practical benchmarks we need significantly fewer iterations, yielding a speed-up of about 20% on average. More importantly though, is the conceptual simplicity of our approach.



(a) A sample MC $\mathcal{D}$.



(b) A sample MDP $\mathcal{M}$.

Fig. 1. Example models.

2 Preliminaries

For a finite set S and vector $x \in \mathbb{R}^{|S|}$, let $x[s] \in \mathbb{R}$ denote the entry of x that corresponds to $s \in S$. Let $S' \subseteq S$ and $a \in \mathbb{R}$. We write $x[S'] = a$ to denote that $x[s] = a$ for all $s \in S'$. Given $x, y \in \mathbb{R}^{|S|}$, $x \leq y$ holds iff $x[s] \leq y[s]$ holds for all $s \in S$. For a function $f\colon \mathbb{R}^{|S|} \to \mathbb{R}^{|S|}$ and $k \geq 0$ we write f^k for the function obtained by applying f k times, i.e., $f^0(x) = x$ and $f^k(x) = f(f^{k-1}(x))$ if $k > 0$.

2.1 Probabilistic Models and Measures

We briefly present probabilistic models and their properties. More details can be found in, e.g., [15].

Definition 1 (Probabilistic Models). *A* Markov Decision Process (MDP) *is a tuple* $\mathcal{M} = (S, Act, \mathbf{P}, s_I, \rho)$*, where*

- *S is a finite set of states, Act is a finite set of actions, s_I is the initial state,*
- *$\mathbf{P}\colon S \times Act \times S \to [0,1]$ is a transition probability function satisfying $\sum_{s' \in S} \mathbf{P}(s, \alpha, s') \in \{0,1\}$ for all $s \in S, \alpha \in Act$, and*
- *$\rho\colon S \times Act \to \mathbb{R}$ is a reward function.*

$\mathcal{M}$ is a Markov Chain (MC) *if $|Act| = 1$.*

Example 1. Figure 1 shows an example MC and an example MDP.

We often simplify notations for MCs by omitting the (unique) action. For an MDP $\mathcal{M} = (S, Act, \mathbf{P}, s_I, \rho)$, the set of *enabled actions* of state $s \in S$ is given by $Act(s) = \{\alpha \in Act \mid \sum_{s' \in S} \mathbf{P}(s, \alpha, s') = 1\}$. We assume that $Act(s) \neq \emptyset$ for each $s \in S$. Intuitively, upon performing action α at state s reward $\rho(s, \alpha)$ is collected and with probability $\mathbf{P}(s, \alpha, s')$ we move to $s' \in S$. Notice that rewards can be positive or negative.

A state $s \in S$ is called *absorbing* if $\mathbf{P}(s, \alpha, s) = 1$ for every $\alpha \in Act(s)$. A *path* of $\mathcal{M}$ is an infinite alternating sequence $\pi = s_0 \alpha_0 s_1 \alpha_1 \ldots$ where $s_i \in S$, $\alpha_i \in$

$\mathit{Act}(s_i)$, and $\mathbf{P}(s_i, \alpha_i, s_{i+1}) > 0$ for all $i \geq 0$. The set of paths of $\mathcal{M}$ is denoted by $\mathit{Paths}^{\mathcal{M}}$. The set of paths that start at $s \in S$ is given by $\mathit{Paths}^{\mathcal{M},s}$. A *finite path* $\hat{\pi} = s_0\alpha_0 \dots \alpha_{n-1}s_n$ is a finite prefix of a path ending with $\mathit{last}(\hat{\pi}) = s_n \in S$. $|\hat{\pi}| = n$ is the length of $\hat{\pi}$, $\mathit{Paths}^{\mathcal{M}}_{\mathit{fin}}$ is the set of finite paths of $\mathcal{M}$, and $\mathit{Paths}^{\mathcal{M},s}_{\mathit{fin}}$ is the set of finite paths that start at state $s \in S$. We consider LTL-like notations for sets of paths. For $k \in \mathbb{N} \cup \{\infty\}$ and $G, H \subseteq S$ let

$$H\,\mathcal{U}^{\leq k}\,G = \{s_0\alpha_0s_1 \dots \in \mathit{Paths}^{\mathcal{M},s_I} \mid s_0, \dots, s_{j-1} \in H,\ s_j \in G \text{ for some } j \leq k\}$$

denote the set of paths that, starting from the initial state s_I, only visit states in H until after at most k steps a state in G is reached. Sets $H\,\mathcal{U}^{>k}\,G$ and $H\,\mathcal{U}^{=k}\,G$ are defined similarly. We use the shorthands $\lozenge^{\leq k}G := S\,\mathcal{U}^{\leq k}\,G$, $\lozenge G := \lozenge^{\leq \infty}G$, and $\square^{\leq k}G := \mathit{Paths}^{\mathcal{M},s_I} \setminus \lozenge^{\leq k}(S \setminus G)$.

A *(deterministic) scheduler* for $\mathcal{M}$ is a function $\sigma\colon \mathit{Paths}^{\mathcal{M}}_{\mathit{fin}} \to \mathit{Act}$ such that $\sigma(\hat{\pi}) \in \mathit{Act}(\mathit{last}(\hat{\pi}))$ for all $\hat{\pi} \in \mathit{Paths}^{\mathcal{M}}_{\mathit{fin}}$. The set of (deterministic) schedulers for $\mathcal{M}$ is $\mathfrak{S}^{\mathcal{M}}$. $\sigma \in \mathfrak{S}^{\mathcal{M}}$ is called *positional* if $\sigma(\hat{\pi})$ only depends on the last state of $\hat{\pi}$, i.e., for all $\hat{\pi}, \hat{\pi}' \in \mathit{Paths}^{\mathcal{M}}_{\mathit{fin}}$ we have $\mathit{last}(\hat{\pi}) = \mathit{last}(\hat{\pi}')$ implies $\sigma(\hat{\pi}) = \sigma(\hat{\pi}')$. For MDP $\mathcal{M}$ and scheduler $\sigma \in \mathfrak{S}^{\mathcal{M}}$ the *probability measure* over finite paths is given by $\mathrm{Pr}^{\mathcal{M},\sigma}_{\mathit{fin}}\colon \mathit{Paths}^{\mathcal{M},s_I}_{\mathit{fin}} \to [0,1]$ with $\mathrm{Pr}^{\mathcal{M},\sigma}_{\mathit{fin}}(s_0 \dots s_n) = \prod_{i=0}^{n-1} \mathbf{P}(s_i, \sigma(s_0 \dots s_i), s_{i+1})$. The probability measure $\mathrm{Pr}^{\mathcal{M},\sigma}$ over measurable sets of infinite paths is obtained via a standard cylinder set construction [15].

Definition 2 (Reachability Probability). *The* reachability probability *of MDP $\mathcal{M} = (S, \mathit{Act}, \mathbf{P}, s_I, \rho)$, $G \subseteq S$, and $\sigma \in \mathfrak{S}^{\mathcal{M}}$ is given by* $\mathrm{Pr}^{\mathcal{M},\sigma}(\lozenge G)$.

For $k \in \mathbb{N} \cup \{\infty\}$, the function $\blacklozenge^{\leq k}G\colon \lozenge G \to \mathbb{R}$ yields the k-bounded reachability reward of a path $\pi = s_0\alpha_0s_1 \dots \in \lozenge G$. We set $\blacklozenge^{\leq k}G(\pi) = \sum_{i=0}^{j-1} \rho(s_i, \alpha_i)$, where $j = \min(\{i \geq 0 \mid s_i \in G\} \cup \{k\})$. We write $\blacklozenge G$ instead of $\blacklozenge^{\leq \infty}G$.

Definition 3 (Expected Reward). *The* expected (reachability) reward *of MDP $\mathcal{M} = (S, \mathit{Act}, \mathbf{P}, s_I, \rho)$, $G \subseteq S$, and $\sigma \in \mathfrak{S}^{\mathcal{M}}$ with $\mathrm{Pr}^{\mathcal{M},\sigma}(\lozenge G) = 1$ is given by the expectation* $\mathbb{E}^{\mathcal{M},\sigma}(\blacklozenge G) = \int_{\pi \in \lozenge G} \blacklozenge G(\pi)\, \mathrm{d}\mathrm{Pr}^{\mathcal{M},\sigma}(\pi)$.

We write $\mathrm{Pr}^{\mathcal{M},\sigma}_s$ and $\mathbb{E}^{\mathcal{M},\sigma}_s$ for the probability measure and expectation obtained by changing the initial state of $\mathcal{M}$ to $s \in S$. If $\mathcal{M}$ is a Markov chain, there is only a single scheduler. In this case we may omit the superscript σ from $\mathrm{Pr}^{\mathcal{M},\sigma}$ and $\mathbb{E}^{\mathcal{M},\sigma}$. We also omit the superscript $\mathcal{M}$ if it is clear from the context. The maximal reachability probability of $\mathcal{M}$ and G is given by $\mathrm{Pr}^{\max}(\lozenge G) = \max_{\sigma \in \mathfrak{S}^{\mathcal{M}}} \mathrm{Pr}^{\sigma}(\lozenge G)$. There is a a positional scheduler that attains this maximum [16]. The same holds for minimal reachability probabilities and maximal or minimal expected rewards.

Example 2. Consider the MDP $\mathcal{M}$ from Fig. 1(b). We are interested in the maximal probability to reach state s_4 given by $\mathrm{Pr}^{\max}(\lozenge\{s_4\})$. Since s_4 is not reachable from s_3 we have $\mathrm{Pr}^{\max}_{s_3}(\lozenge\{s_4\}) = 0$. Intuitively, choosing action β at state s_0 makes reaching s_3 more likely, which should be avoided in order to maximize the probability to reach s_4. We therefore assume a scheduler σ that always chooses action α at state s_0. Starting from the initial state s_0, we then eventually take the transition from s_2 to s_3 or the transition from s_2 to s_4 with probability one. The resulting probability to reach s_4 is given by $\mathrm{Pr}^{\max}(\lozenge\{s_4\}) = \mathrm{Pr}^{\sigma}(\lozenge\{s_4\}) = 0.3/(0.1 + 0.3) = 0.75$.

2.2 Probabilistic Model Checking via Interval Iteration

In the following we present approaches to compute reachability probabilities and expected rewards. We consider approximative computations. Exact computations are handled in e.g. [17,18] For the sake of clarity, we focus on reachability probabilities and sketch how the techniques can be lifted to expected rewards.

Reachability Probabilities. We fix an MDP $\mathcal{M} = (S, Act, \mathbf{P}, s_I, \rho)$, a set of goal states $G \subseteq S$, and a precision parameter $\varepsilon > 0$.

Problem 1. Compute an ε-approximation of the maximal reachability probability $\mathrm{Pr}^{\max}(\lozenge G)$, i.e., compute a value $r \in [0, 1]$ with $|r - \mathrm{Pr}^{\max}(\lozenge G)| < \varepsilon$.

We briefly sketch how to compute such a value r via *interval iteration* [12,13,19]. The computation for minimal reachability probabilities is analogous.

W.l.o.g. it is assumed that the states in G are absorbing. Using graph algorithms, we compute $S_0 = \{s \in S \mid \mathrm{Pr}_s^{\max}(\lozenge G) = 0\}$ and partition the state space of $\mathcal{M}$ into $S = S_0 \uplus G \uplus S_?$ with $S_? = S \setminus (G \cup S_0)$. If $s_I \in S_0$ or $s_I \in G$, the probability $\mathrm{Pr}^{\max}(\lozenge G)$ is 0 or 1, respectively. From now on we assume $s_I \in S_?$.

We say that $\mathcal{M}$ is *contracting* with respect to $S' \subseteq S$ if $\mathrm{Pr}_s^{\sigma}(\lozenge S') = 1$ for all $s \in S$ and for all $\sigma \in \mathfrak{S}^{\mathcal{M}}$. We assume that $\mathcal{M}$ is contracting with respect to $G \cup S_0$. Otherwise, we apply a transformation on the so-called *end components*[1] of $\mathcal{M}$, yielding a contracting MDP $\mathcal{M}'$ with the same maximal reachability probability as $\mathcal{M}$. Roughly, this transformation replaces each end component of $\mathcal{M}$ with a single state whose enabled actions coincide with the actions that previously lead outside of the end component. This step is detailed in [13,19].

We have $x^*[s] = \mathrm{Pr}_s^{\max}(\lozenge G)$ for $s \in S$ and the unique fixpoint x^* of the function $f\colon \mathbb{R}^{|S|} \to \mathbb{R}^{|S|}$ with $f(x)[S_0] = 0$, $f(x)[G] = 1$, and

$$f(x)[s] = \max_{\alpha \in Act(s)} \sum_{s' \in S} \mathbf{P}(s, \alpha, s') \cdot x[s']$$

for $s \in S_?$. Hence, computing $\mathrm{Pr}^{\max}(\lozenge G)$ reduces to finding the fixpoint of f.

A popular technique for this purpose is the *value iteration* algorithm [1]. Given a starting vector $x \in \mathbb{R}^{|S|}$ with $x[S_0] = 0$ and $x[G] = 1$, standard value iteration computes $f^k(x)$ for increasing k until $\max_{s \in S} |f^k(x)[s] - f^{k-1}(x)[s]| < \varepsilon$ holds for a predefined precision $\varepsilon > 0$. As pointed out in, e.g., [13], there is no

[1] Intuitively, an end component is a set of states $S' \subseteq S$ such that there is a scheduler inducing that from any $s \in S'$ exactly the states in S' are visited infinitely often.

guarantee on the preciseness of the result $r = f^k(x)[s_I]$, i.e., standard value iteration does not give any evidence on the error $|r - \Pr^{\max}(\Diamond G)|$. The intuitive reason is that value iteration only approximates the fixpoint x^* from one side, yielding no indication on the distance between the current result and x^*.

Example 3. Consider the MDP $\mathcal{M}$ from Fig. 1(b). We invoked standard value iteration in `PRISM` [7] and `Storm` [8] to compute the reachability probability $\Pr^{\max}(\Diamond\{s_4\})$. Recall from Example 2 that the correct solution is 0.75. With (absolute) precision $\varepsilon = 10^{-6}$ both model checkers returned 0.7248. Notice that the user can improve the precision by considering, e.g., $\varepsilon = 10^{-8}$ which yields 0.7497. However, there is no guarantee on the preciseness of a given result.

The *interval iteration* algorithm [12,13,19] addresses the impreciseness of value iteration. The idea is to approach the fixpoint x^* from below and from above. The first step is to find starting vectors $x_\ell, x_u \in \mathbb{R}^{|S|}$ satisfying $x_\ell[S_0] = x_u[S_0] = 0$, $x_\ell[G] = x_u[G] = 1$, and $x_\ell \le x^* \le x_u$. As the entries of x^* are probabilities, it is always valid to set $x_\ell[S_?] = 0$ and $x_u[S_?] = 1$. We have $f^k(x_\ell) \le x^* \le f^k(x_u)$ for any $k \ge 0$. Interval iteration computes $f^k(x_\ell)$ and $f^k(x_u)$ for increasing k until $\max_{s\in S} |f^k(x_\ell)[s] - f^k(x_u)[s]| < 2\varepsilon$. For the result $r = 1/2 \cdot (f^k(x_\ell)[s_I] + f^k(x_u)[s_I])$ we obtain that $|r - \Pr^{\max}(\Diamond G)| < \varepsilon$, i.e., we get a sound approximation of the maximal reachability probability.

Example 4. We invoked interval iteration in `PRISM` and `Storm` to compute the reachability probability $\Pr^{\max}(\Diamond\{s_4\})$ for the MDP $\mathcal{M}$ from Fig. 1(b). Both implementations correctly yield an ε-approximation of $\Pr^{\max}(\Diamond\{s_4\})$, where we considered $\varepsilon = 10^{-6}$. However, both `PRISM` and `Storm` required roughly 300,000 iterations for convergence.

Expected Rewards. Whereas [13,19] only consider reachability probabilities, [12] extends interval iteration to compute expected rewards. Let $\mathcal{M}$ be an MDP and G be a set of absorbing states such that $\mathcal{M}$ is contracting with respect to G.

Problem 2. Compute an ε-approximation of the maximal expected reachability reward $\mathbb{E}^{\max}(\blacklozenge G)$, i.e., compute a value $r \in \mathbb{R}$ with $|r - \mathbb{E}^{\max}(\blacklozenge G)| < \varepsilon$.

We have $x^*[s] = \mathbb{E}^{\max}_s(\blacklozenge G)$ for the unique fixpoint x^* of $g\colon \mathbb{R}^{|S|} \to \mathbb{R}^{|S|}$ with

$$g(x)[G] = 0 \quad\text{and}\quad g(x)[s] = \max_{\alpha\in Act(s)} \rho(s,\alpha) + \sum_{s'\in S} \mathbf{P}(s,\alpha,s') \cdot x[s']$$

for $s \notin G$. As for reachability probabilities, interval iteration can be applied to approximate this fixpoint. The crux lies in finding appropriate starting vectors $x_\ell, x_u \in \mathbb{R}^{|S|}$ guaranteeing $x_\ell \le x^* \le x_u$. To this end, [12] describe graph based algorithms that give an upper bound on the expected number of times each individual state $s \in S \setminus G$ is visited. This then yields an approximation of the expected amount of reward collected at the various states.

3 Sound Value Iteration for MCs

We present an algorithm for computing reachability probabilities and expected rewards as in Problems 1 and 2. The algorithm is an alternative to the interval iteration approach [12,20] but (i) does not require an a priori computation of starting vectors $x_\ell, x_u \in \mathbb{R}^{|S|}$ and (ii) converges faster on many practical benchmarks as shown in Sect. 5. For the sake of simplicity, we first restrict to computing reachability probabilities on MCs.

In the following, let $\mathcal{D} = (S, \mathbf{P}, s_I, \rho)$ be an MC, $G \subseteq S$ be a set of absorbing goal states and $\varepsilon > 0$ be a precision parameter. We consider the partition $S = S_0 \uplus G \uplus S_?$ as in Sect. 2.2. The following theorem captures the key insight of our algorithm.

Theorem 1. *For MC $\mathcal{D}$ let G and $S_?$ be as above and $k \geq 0$ with $\Pr_s(\Box^{\leq k} S_?) < 1$ for all $s \in S_?$. We have*

$$\Pr(\Diamond^{\leq k} G) + \Pr(\Box^{\leq k} S_?) \cdot \min_{s \in S_?} \frac{\Pr_s(\Diamond^{\leq k} G)}{1 - \Pr_s(\Box^{\leq k} S_?)}$$
$$\leq \Pr(\Diamond G) \leq \Pr(\Diamond^{\leq k} G) + \Pr(\Box^{\leq k} S_?) \cdot \max_{s \in S_?} \frac{\Pr_s(\Diamond^{\leq k} G)}{1 - \Pr_s(\Box^{\leq k} S_?)}.$$

Theorem 1 allows us to approximate $\Pr(\Diamond G)$ by computing for increasing $k \in \mathbb{N}$

- $\Pr(\Diamond^{\leq k} G)$, the probability to reach a state in G within k steps, and
- $\Pr(\Box^{\leq k} S_?)$, the probability to stay in $S_?$ during the first k steps.

This can be realized via a value-iteration based procedure. The obtained bounds on $\Pr(\Diamond G)$ can be tightened arbitrarily since $\Pr(\Box^{\leq k} S_?)$ approaches 0 for increasing k. In the following, we address the correctness of Theorem 1, describe the details of our algorithm, and indicate how the results can be lifted to expected rewards.

3.1 Approximating Reachability Probabilities

To approximate the reachability probability $\Pr(\Diamond G)$, we consider the step bounded reachability probability $\Pr(\Diamond^{\leq k} G)$ for $k \geq 0$ and provide a lower and an upper bound for the 'missing' probability $\Pr(\Diamond G) - \Pr(\Diamond^{\leq k} G)$. Note that $\Diamond G$ is the disjoint union of the paths that reach G *within* k steps (given by $\Diamond^{\leq k} G$) and the paths that reach G only *after* k steps (given by $S_? \,\mathcal{U}^{>k}\, G$).

Lemma 1. *For any $k \geq 0$ we have $\Pr(\Diamond G) = \Pr(\Diamond^{\leq k} G) + \Pr(S_? \,\mathcal{U}^{>k}\, G)$.*

A path $\pi \in S_? \,\mathcal{U}^{>k}\, G$ reaches some state $s \in S_?$ after *exactly* k steps. This yields the partition $S_? \,\mathcal{U}^{>k}\, G = \biguplus_{s \in S_?} (S_? \,\mathcal{U}^{=k} \{s\} \cap \Diamond G)$. It follows that

$$\Pr(S_? \,\mathcal{U}^{>k}\, G) = \sum_{s \in S_?} \Pr(S_? \,\mathcal{U}^{=k}\, \{s\}) \cdot \Pr_s(\Diamond G).$$

Consider $\ell, u \in [0,1]$ with $\ell \leq \mathrm{Pr}_s(\Diamond G) \leq u$ for all $s \in S_?$, i.e., ℓ and u are lower and upper bounds for the reachability probabilities within $S_?$. We have

$$\sum_{s \in S_?} \mathrm{Pr}(S_? \,\mathcal{U}^{=k} \{s\}) \cdot \mathrm{Pr}_s(\Diamond G) \leq \sum_{s \in S_?} \mathrm{Pr}(S_? \,\mathcal{U}^{=k} \{s\}) \cdot u = \mathrm{Pr}(\Box^{\leq k} S_?) \cdot u.$$

We can argue similar for the lower bound ℓ. With *Lemma* 1 we get the following.

Proposition 1. *For MC $\mathcal{D}$ with G, $S_?$, ℓ, u as above and any $k \geq 0$ we have*

$$\mathrm{Pr}(\Diamond^{\leq k} G) + \mathrm{Pr}(\Box^{\leq k} S_?) \cdot \ell \leq \mathrm{Pr}(\Diamond G) \leq \mathrm{Pr}(\Diamond^{\leq k} G) + \mathrm{Pr}(\Box^{\leq k} S_?) \cdot u.$$

Remark 1. The bounds for $\mathrm{Pr}(\Diamond G)$ given by Proposition 1 are similar to the bounds obtained after performing k iterations of interval iteration with starting vectors $x_\ell, x_u \in \mathbb{R}^{|S|}$, where $x_\ell[S_?] = \ell$ and $x_u[S_?] = u$.

We now discuss how the bounds ℓ and u can be obtained from the step bounded probabilities $\mathrm{Pr}_s(\Diamond^{\leq k} G)$ and $\mathrm{Pr}_s(\Box^{\leq k} S_?)$ for $s \in S_?$. We focus on the upper bound u. The reasoning for the lower bound ℓ is similar.

Let $s_{\max} \in S_?$ be a state with maximal reachability probability, that is $s_{\max} \in \arg\max_{\mathrm{s} \in \mathrm{S}_?} \mathrm{Pr_s}(\Diamond \mathrm{G})$. From Proposition 1 we get

$$\mathrm{Pr}_{s_{\max}}(\Diamond G) \leq \mathrm{Pr}_{s_{\max}}(\Diamond^{\leq k} G) + \mathrm{Pr}_{s_{\max}}(\Box^{\leq k} S_?) \cdot \mathrm{Pr}_{s_{\max}}(\Diamond G).$$

We solve the inequality for $\mathrm{Pr}_{s_{\max}}(\Diamond G)$ (assuming $\mathrm{Pr}_s(\Box^{\leq k} S_?) < 1$ for all $s \in S_?$):

$$\mathrm{Pr}_{s_{\max}}(\Diamond G) \leq \frac{\mathrm{Pr}_{s_{\max}}(\Diamond^{\leq k} G)}{1 - \mathrm{Pr}_{s_{\max}}(\Box^{\leq k} S_?)} \leq \max_{s \in S_?} \frac{\mathrm{Pr}_s(\Diamond^{\leq k} G)}{1 - \mathrm{Pr}_s(\Box^{\leq k} S_?)}.$$

Proposition 2. *For MC $\mathcal{D}$ let G and $S_?$ be as above and $k \geq 0$ such that $\mathrm{Pr}_s(\Box^{\leq k} S_?) < 1$ for all $s \in S_?$. For every $\hat{s} \in S_?$ we have*

$$\min_{s \in S_?} \frac{\mathrm{Pr}_s(\Diamond^{\leq k} G)}{1 - \mathrm{Pr}_s(\Box^{\leq k} S_?)} \leq \mathrm{Pr}_{\hat{s}}(\Diamond G) \leq \max_{s \in S_?} \frac{\mathrm{Pr}_s(\Diamond^{\leq k} G)}{1 - \mathrm{Pr}_s(\Box^{\leq k} S_?)}.$$

Theorem 1 is a direct consequence of Propositions 1 and 2.

3.2 Extending the Value Iteration Approach

Recall the standard value iteration algorithm for approximating $\mathrm{Pr}(\Diamond G)$ as discussed in Sect. 2.2. The function $f \colon \mathbb{R}^{|S|} \to \mathbb{R}^{|S|}$ for MCs simplifies to $f(x)[S_0] = 0$, $f(x)[G] = 1$, and $f(x)[s] = \sum_{s' \in S} \mathbf{P}(s, s') \cdot x[s']$ for $s \in S_?$. We can compute the k-step bounded reachability probability at every state $s \in S$

```
    Input  : MC $\mathcal{D} = (S, \mathbf{P}, s_I, \rho)$, absorbing states $G \subseteq S$, precision $\varepsilon > 0$
    Output : $r \in \mathbb{R}$ with $|r - \Pr(\Diamond G)| < \varepsilon$
 1  $S_? \leftarrow S \setminus \big(\{s \in S \mid \Pr_s(\Diamond G) = 0\} \cup G\big)$
 2  initialize $x_0, y_0 \in \mathbb{R}^{|S|}$ with $x_0[G] = 1$, $x_0[S \setminus G] = 0$, $y_0[S_?] = 1$, $y_0[S \setminus S_?] = 0$
 3  $\ell_0 \leftarrow -\infty$; $u_0 \leftarrow +\infty$
 4  $k \leftarrow 0$
 5  repeat
 6      $k \leftarrow k + 1$
 7      $x_k \leftarrow f(x_{k-1})$; $y_k \leftarrow h(y_{k-1})$
 8      if $y_k[s] < 1$ for all $s \in S_?$ then
 9          $\ell_k \leftarrow \max(\ell_{k-1}, \min_{s \in S_?} \frac{x_k[s]}{1 - y_k[s]})$; $u_k \leftarrow \min(u_{k-1}, \max_{s \in S_?} \frac{x_k[s]}{1 - y_k[s]})$
10  until $y_k[s_I] \cdot (u_k - \ell_k) < 2 \cdot \varepsilon$
11  return $x_k[s_I] + y_k[s_I] \cdot \frac{\ell_k + u_k}{2}$
```

Algorithm 1: Sound value iteration for MCs.

by performing k iterations of value iteration [15, Remark 10.104]. More precisely, when applying f k times on starting vector $x \in \mathbb{R}^{|S|}$ with $x[G] = 1$ and $x[S \setminus G] = 0$ we get $\Pr_s(\Diamond^{\leq k} G) = f^k(x)[s]$. The probabilities $\Pr_s(\Box^{\leq k} S_?)$ for $s \in S$ can be computed similarly. Let $h \colon \mathbb{R}^{|S|} \to \mathbb{R}^{|S|}$ with $h(y)[S \setminus S_?] = 0$ and $h(y)[s] = \sum_{s' \in S} \mathbf{P}(s, s') \cdot y[s']$ for $s \in S_?$. For starting vector $y \in \mathbb{R}^{|S|}$ with $y[S_?] = 1$ and $y[S \setminus S_?] = 0$ we get $\Pr_s(\Box^{\leq k} S_?) = h^k(y)[s]$.

Algorithm 1 depicts our approach. It maintains vectors $x_k, y_k \in \mathbb{R}^{|S|}$ which, after k iterations of the loop, store the k-step bounded probabilities $\Pr_s(\Diamond^{\leq k} G)$ and $\Pr_s(\Box^{\leq k} S_?)$, respectively. Additionally, the algorithm considers lower bounds ℓ_k and upper bounds u_k such that the following invariant holds.

Lemma 2. *After executing the loop of Algorithm 1 k times we have for all $s \in S_?$ that $x_k[s] = \Pr_s(\Diamond^{\leq k} G)$, $y_k[s] = \Pr_s(\Box^{\leq k} S_?)$, and $\ell_k \leq \Pr_s(\Diamond G) \leq u_k$.*

The correctness of the algorithm follows from Theorem 1. Termination is guaranteed since $\Pr(\Diamond(S_0 \cup G)) = 1$ and therefore $\lim_{k \to \infty} \Pr(\Box^{\leq k} S_?) = \Pr(\Box S_?) = 0$.

Theorem 2. *Algorithm 1 terminates for any MC $\mathcal{D}$, goal states G, and precision $\varepsilon > 0$. The returned value r satisfies $|r - \Pr(\Diamond G)| < \varepsilon$.*

Example 5. We apply Algorithm 1 for the MC in Fig. 1(a) and the set of goal states $G = \{s_4\}$. We have $S_? = \{s_0, s_1, s_2\}$. After $k = 3$ iterations it holds that

$$x_3[s_0] = 0.00003 \quad x_3[s_1] = 0.003 \quad x_3[s_2] = 0.3$$
$$y_3[s_0] = 0.99996 \quad y_3[s_1] = 0.996 \quad y_3[s_2] = 0.6$$

Hence, $\frac{x_3[s]}{1 - y_3[s]} = \frac{3}{4} = 0.75$ for all $s \in S_?$. We get $\ell_3 = u_3 = 0.75$. The algorithm converges for any $\varepsilon > 0$ and returns the correct solution $x_3[s_0] + y_3[s_0] \cdot 0.75 = 0.75$.

3.3 Sound Value Iteration for Expected Rewards

We lift our approach to expected rewards in a straightforward manner. Let $G \subseteq S$ be a set of absorbing goal states of MC $\mathcal{D}$ such that $\Pr(\Diamond G) = 1$. Further let $S_? = S \setminus G$. For $k \geq 0$ we observe that the expected reward $\mathbb{E}(\blacklozenge G)$ can be split into the expected reward collected within k steps and the expected reward collected only after k steps, i.e., $\mathbb{E}(\blacklozenge G) = \mathbb{E}(\blacklozenge^{\leq k} G) + \sum_{s \in S_?} \Pr(S_? \mathcal{U}^{=k} \{s\}) \cdot \mathbb{E}_s(\blacklozenge G)$. Following a similar reasoning as in Sect. 3.1 we can show the following.

Theorem 3. *For MC $\mathcal{D}$ let G and $S_?$ be as before and $k \geq 0$ such that $\Pr_s(\Box^{\leq k} S_?) < 1$ for all $s \in S_?$. We have*

$$\mathbb{E}(\blacklozenge^{\leq k} G) + \Pr(\Box^{\leq k} S_?) \cdot \min_{s \in S_?} \frac{\mathbb{E}_s(\blacklozenge^{\leq k} G)}{1 - \Pr_s(\Box^{\leq k} S_?)}$$
$$\leq \mathbb{E}(\blacklozenge G) \leq \mathbb{E}(\blacklozenge^{\leq k} G) + \Pr(\Box^{\leq k} S_?) \cdot \max_{s \in S_?} \frac{\mathbb{E}_s(\blacklozenge^{\leq k} G)}{1 - \Pr_s(\Box^{\leq k} S_?)}.$$

Recall the function $g \colon \mathbb{R}^{|S|} \to \mathbb{R}^{|S|}$ from Sect. 2.2, given by $g(x)[G] = 0$ and $g(x)[s] = \rho(s) + \sum_{s' \in S} \mathbf{P}(s, s') \cdot x[s']$ for $s \in S_?$. For $s \in S$ and $x \in \mathbb{R}^{|S|}$ with $x[S] = 0$ we have $\mathbb{E}_s(\blacklozenge^{\leq k} G) = g^k(x)[s]$. We modify Algorithm 1 such that it considers function g instead of function f. Then, the returned value r satisfies $|r - \mathbb{E}(\blacklozenge G)| < \varepsilon$.

3.4 Optimizations

Algorithm 1 can make use of *initial bounds* $\ell_0, u_0 \in \mathbb{R}$ with $\ell_0 \leq \Pr_s(\Diamond G) \leq u_0$ for all $s \in S_?$. Such bounds could be derived, e.g., from domain knowledge or during preprocessing [12]. The algorithm always chooses the largest available lower bound for ℓ_k and the lowest available upper bound for u_k, respectively. If Algorithm 1 and interval iteration are initialized with the same bounds, Algorithm 1 always requires as most as many iterations compared to interval iteration (cf. Remark 1).

Gauss-Seidel value iteration [1,12] is an optimization for standard value iteration and interval iteration that potentially leads to faster convergence. When computing $f(x)[s]$ for $s \in S_?$, the idea is to consider already computed results $f(x)[s']$ from the current iteration. Formally, let $\prec \subseteq S \times S$ be some strict total ordering of the states. Gauss-Seidel value iteration considers instead of function f the function $f_\prec \colon \mathbb{R}^{|S|} \to \mathbb{R}^{|S|}$ with $f_\prec[S_0] = 0$, $f_\prec[G] = 1$, and

$$f_\prec(x)[s] = \sum_{s' \prec s} \mathbf{P}(s, s') \cdot f_\prec(x)[s'] + \sum_{s' \not\prec s} \mathbf{P}(s, s') \cdot x[s'].$$

Values $f_\prec(x)[s]$ for $s \in S$ are computed in the order defined by $\prec$. This idea can also be applied to our approach. To this end, we replace f by $f_\prec$ and h by $h_\prec$, where $h_\prec$ is defined similarly. More details are given in [21].

Topological value iteration [14] employs the graphical structure of the MC $\mathcal{D}$. The idea is to decompose the states S of $\mathcal{D}$ into strongly connected components[2]

[2] $S' \subseteq S$ is a connected component if s can be reached from s' for all $s, s' \in S'$. S' is a strongly connected component if no superset of S' is a connected component.

(SCCs) that are analyzed individually. The procedure can improve the runtime of classical value iteration since for a single iteration only the values for the current SCC have to be updated. A topological variant of interval iteration is introduced in [12]. Given these results, sound value iteration can be extended similarly.

4 Sound Value Iteration for MDPs

We extend sound value iteration to compute reachability probabilities in MDPs. Assume an MDP $\mathcal{M} = (S, Act, \mathbf{P}, s_I, \rho)$ and a set of absorbing goal states G. For simplicity, we focus on maximal reachability probabilities, i.e., we compute $\Pr^{\max}(\Diamond G)$. Minimal reachability probabilities and expected rewards are analogous. As in Sect. 2.2 we consider the partition $S = S_0 \uplus G \uplus S_?$ such that $\mathcal{M}$ is contracting with respect to $G \cup S_0$.

4.1 Approximating Maximal Reachability Probabilities

We argue that our results for MCs also hold for MDPs under a given scheduler $\sigma \in \mathfrak{S}^{\mathcal{M}}$. Let $k \geq 0$ such that $\Pr^{\sigma}_s(\Box^{\leq k} S_?) < 1$ for all $s \in S_?$. Following the reasoning as in Sect. 3.1 we get

$$\Pr^{\sigma}(\Diamond^{\leq k} G) + \Pr^{\sigma}(\Box^{\leq k} S_?) \cdot \min_{s \in S_?} \frac{\Pr^{\sigma}_s(\Diamond^{\leq k} G)}{1 - \Pr^{\sigma}_s(\Box^{\leq k} S_?)} \leq \Pr^{\sigma}(\Diamond G) \leq \Pr^{\max}(\Diamond G).$$

Next, assume an upper bound $u \in \mathbb{R}$ with $\Pr^{\max}_s(\Diamond G) \leq u$ for all $s \in S_?$. For a scheduler $\sigma_{\max} \in \mathfrak{S}^{\mathcal{M}}$ that attains the maximal reachability probability, i.e., $\sigma_{\max} \in \arg\max_{\sigma \in \mathfrak{S}^{\mathcal{M}}} \Pr^{\sigma}(\Diamond \mathrm{G})$ it holds that

$$\begin{aligned}\Pr^{\max}(\Diamond G) = \Pr^{\sigma_{\max}}(\Diamond G) &\leq \Pr^{\sigma_{\max}}(\Diamond^{\leq k} G) + \Pr^{\sigma_{\max}}(\Box^{\leq k} S_?) \cdot u \\ &\leq \max_{\sigma \in \mathfrak{S}^{\mathcal{M}}} \big(\Pr^{\sigma}(\Diamond^{\leq k} G) + \Pr^{\sigma}(\Box^{\leq k} S_?) \cdot u\big).\end{aligned}$$

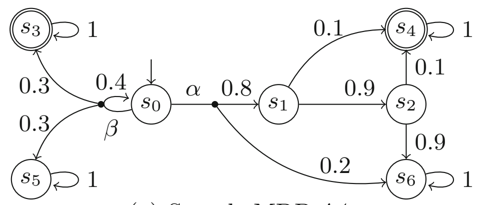
We obtain the following theorem which is the basis of our algorithm.

Theorem 4. *For MDP $\mathcal{M}$ let G, $S_?$, and u be as above. Assume $\sigma \in \mathfrak{S}^{\mathcal{M}}$ and $k \geq 0$ such that $\sigma \in \arg\max_{\sigma' \in \mathfrak{S}^{\mathcal{M}}} \Pr^{\sigma'}(\Diamond^{\leq \mathrm{k}} \mathrm{G}) + \Pr^{\sigma'}(\Box^{\leq \mathrm{k}} \mathrm{S}_?) \cdot \mathrm{u}$ and $\Pr^{\sigma}_s(\Box^{\leq k} S_?) < 1$ for all $s \in S_?$. We have*

$$\begin{aligned}&\Pr^{\sigma}(\Diamond^{\leq k} G) + \Pr^{\sigma}(\Box^{\leq k} S_?) \cdot \min_{s \in S_?} \frac{\Pr^{\sigma}_s(\Diamond^{\leq k} G)}{1 - \Pr^{\sigma}_s(\Box^{\leq k} S_?)} \\ &\leq \Pr^{\max}(\Diamond G) \leq \Pr^{\sigma}(\Diamond^{\leq k} G) + \Pr^{\sigma}(\Box^{\leq k} S_?) \cdot u.\end{aligned}$$

Similar to the results for MCs it also holds that $\Pr^{\max}(\Diamond G) \leq \max_{\sigma \in \mathfrak{S}^{\mathcal{M}}} \hat{u}^{\sigma}_k$ with

$$\hat{u}^{\sigma}_k := \Pr^{\sigma}(\Diamond^{\leq k} G) + \Pr^{\sigma}(\Box^{\leq k} S_?) \cdot \max_{s \in S_?} \frac{\Pr^{\sigma}_s(\Diamond^{\leq k} G)}{1 - \Pr^{\sigma}_s(\Box^{\leq k} S_?)}.$$



(a) Sample MDP $\mathcal{M}$.

	$\mathrm{Pr}_{s_0}^{\sigma_\alpha}$	$\mathrm{Pr}_{s_0}^{\sigma_{\beta\alpha}}$	$\mathrm{Pr}_{s_0}^{\sigma_{\beta\beta}}$	$\mathrm{Pr}_{s_1}^{\sigma}$	$\mathrm{Pr}_{s_2}^{\sigma}$
$\lozenge^{\leq 1}G$	0	0.3	0.3	0.1	0.1
$\square^{\leq 1}S_?$	0.8	0.4	0.4	0.9	0
$\lozenge^{\leq 2}G$	0.1	0.3	0.42	0.1	0.1
$\square^{\leq 2}S_?$	0.72	0.32	0.16	0	0

(b) Step bounded probabilities for $\mathcal{M}$.

Fig. 2. Example MDP with corresponding step bounded probabilities.

However, this upper bound can not trivially be embedded in a value iteration based procedure. Intuitively, in order to compute the upper bound for iteration k, one can not necessarily build on the results for iteration $k-1$.

Example 6. Consider the MDP $\mathcal{M}$ given in Fig. 2(a). Let $G = \{s_3, s_4\}$ be the set of goal states. We therefore have $S_? = \{s_0, s_1, s_2\}$. In Fig. 2(b) we list step bounded probabilities with respect to the possible schedulers, where σ_α, $\sigma_{\beta\alpha}$, and $\sigma_{\beta\beta}$ refer to schedulers with $\sigma_\alpha(s_0) = \alpha$ and for $\gamma \in \{\alpha, \beta\}$, $\sigma_{\beta\gamma}(s_0) = \beta$ and $\sigma_{\beta\gamma}(s_0 \beta s_0) = \gamma$. Notice that the probability measures $\mathrm{Pr}_{s_1}^{\sigma}$ and $\mathrm{Pr}_{s_2}^{\sigma}$ are independent of the considered scheduler σ. For step bounds $k \in \{1, 2\}$ we get

- $\max_{\sigma \in \mathfrak{S}^{\mathcal{M}}} \hat{u}_1^{\sigma} = \hat{u}_1^{\sigma_\alpha} = 0 + 0.8 \cdot \max(0, 1, 0) = 0.8$ and
- $\max_{\sigma \in \mathfrak{S}^{\mathcal{M}}} \hat{u}_2^{\sigma} = \hat{u}_2^{\sigma_{\beta\beta}} = 0.42 + 0.16 \cdot \max(0.5, 0.19, 1) = 0.5$.

4.2 Extending the Value Iteration Approach

The idea of our algorithm is to compute the bounds for $\mathrm{Pr}^{\max}(\lozenge G)$ as in Theorem 4 for increasing $k \geq 0$. Algorithm 2 outlines the procedure. Similar to Algorithm 1 for MCs, vectors $x_k, y_k \in \mathbb{R}^{|S|}$ store the step bounded probabilities $\mathrm{Pr}_s^{\sigma_k}(\lozenge^{\leq k} G)$ and $\mathrm{Pr}_s^{\sigma_k}(\square^{\leq k} S_?)$ for any $s \in S$. In addition, schedulers σ_k and upper bounds $u_k \geq \max_{s \in S_?} \mathrm{Pr}_s^{\max}(\lozenge G)$ are computed in a way that Theorem 4 is applicable.

Lemma 3. *After executing k iterations of Algorithm 2 we have for all $s \in S_?$ that $x_k[s] = \mathrm{Pr}_s^{\sigma_k}(\lozenge^{\leq k} G)$, $y_k[s] = \mathrm{Pr}_s^{\sigma_k}(\square^{\leq k} S_?)$, and $\ell_k \leq \mathrm{Pr}_s^{\max}(\lozenge G) \leq u_k$, where $\sigma_k \in \arg\max_{\sigma \in \mathfrak{S}^{\mathcal{M}}} \mathrm{Pr}_{\mathrm{s}}^{\sigma}(\lozenge^{\leq \mathrm{k}} \mathrm{G}) + \mathrm{Pr}_{\mathrm{s}}^{\sigma}(\square^{\leq \mathrm{k}} \mathrm{S}_?) \cdot \mathrm{u}_{\mathrm{k}}$.*

The lemma holds for $k = 0$ as x_0, y_0, and u_0 are initialized accordingly. For $k > 0$ we assume that the claim holds after $k-1$ iterations, i.e., for x_{k-1}, y_{k-1}, u_{k-1} and scheduler σ_{k-1}. The results of the kth iteration are obtained as follows.

The function *findAction* illustrated in Algorithm 3 determines the choices of a scheduler $\sigma_k \in \arg\max_{\sigma \in \mathfrak{S}^{\mathcal{M}}} \mathrm{Pr}_{\mathrm{s}}^{\sigma}(\lozenge^{\leq \mathrm{k}} \mathrm{G}) + \mathrm{Pr}_{\mathrm{s}}^{\sigma}(\square^{\leq \mathrm{k}} \mathrm{S}_?) \cdot \mathrm{u}_{\mathrm{k}-1}$ for $s \in S_?$. The idea is to consider at state s an action $\sigma_k(s) = \alpha \in \mathit{Act}(s)$ that maximizes

$$\mathrm{Pr}_s^{\sigma_k}(\lozenge^{\leq k} G) + \mathrm{Pr}_s^{\sigma_k}(\square^{\leq k} S_?) \cdot u_{k-1} = \sum_{s' \in S} \mathbf{P}(s, \alpha, s') \cdot (x_{k-1}[s'] + y_{k-1}[s'] \cdot u_{k-1}).$$

```
**Input**  : MDP $\mathcal{M} = (S, Act, \mathbf{P}, s_I, \rho)$, absorbing states $G \subseteq S$, precision $\varepsilon > 0$
**Output** : $r \in \mathbb{R}$ with $|r - \Pr^{\max}(\lozenge G)| < \varepsilon$
1  $S_0 \leftarrow \{s \in S \mid \Pr^{\max}_s(\lozenge G) = 0\}$
2  assert that $\mathcal{M}$ is contracting with respect to $G \cup S_0$
3  $S_? \leftarrow S \setminus (S_0 \cup G)$
4  initialize $x_0, y_0 \in \mathbb{R}^{|S|}$ with $x_0[G] = 1$, $x_0[S \setminus G] = 0$, $y_0[S_?] = 1$, $y_0[S \setminus S_?] = 0$
5  $\ell_0 \leftarrow -\infty$; $u_0 \leftarrow +\infty$; $d_0 \leftarrow -\infty$
6  $k \leftarrow 0$
7  **repeat**
8      $k \leftarrow k + 1$
9      initialize $x_k, y_k \in \mathbb{R}^{|S|}$ with $x_k[G] = 1$, $x_k[S_0] = 0$, $y_k[S \setminus S_?] = 0$
10     $d_k \leftarrow d_{k-1}$
11     **foreach** $s \in S_?$ **do**
12         $\alpha \leftarrow \mathit{findAction}(x_{k-1}, y_{k-1}, s, u_{k-1})$
13         $d_k \leftarrow \max(d_k, \mathit{decisionValue}(x_{k-1}, y_{k-1}, s, \alpha))$
14         $x_k[s] \leftarrow \sum_{s' \in S} \mathbf{P}(s, \alpha, s') \cdot x_{k-1}[s']$
15         $y_k[s] \leftarrow \sum_{s' \in S} \mathbf{P}(s, \alpha, s') \cdot y_{k-1}[s']$
16     **if** $y_k[s] < 1$ for all $s \in S_?$ **then**
17         $\ell_k \leftarrow \max(\ell_{k-1}, \min_{s \in S_?} \frac{x_k[s]}{1 - y_k[s]})$
18         $u_k \leftarrow \min(u_{k-1}, \max(d_k, \max_{, \in S_?} \frac{x_k[s]}{1 - y_k[s]}))$
19 **until** $y_k[s_I] \cdot (u_k - \ell_k) < 2 \cdot \varepsilon$
20 **return** $x_k[s_I] + y_k[s_I] \cdot \frac{\ell_k + u_k}{2}$
```

Algorithm 2: Sound value iteration for MDPs

For the case where no real upper bound is known (i.e., $u_{k-1} = \infty$) we implicitly assume a sufficiently large value for u_{k-1} such that $\Pr^{\sigma}_s(\lozenge^{\leq k} G)$ becomes negligible. Upon leaving state s, σ_k mimics σ_{k-1}, i.e., we set $\sigma_k(s\alpha s_1 \alpha_1 \dots s_n) = \sigma_{k-1}(s_1 \alpha_1 \dots s_n)$. After executing Line 15 of Algorithm 2 we have $x_k[s] = \Pr^{\sigma_k}_s(\lozenge^{\leq k} G)$ and $y_k[s] = \Pr^{\sigma_k}_s(\square^{\leq k} S_?)$.

It remains to derive an upper bound u_k. To ensure that Lemma 3 holds we require (i) $u_k \geq \max_{s \in S_?} \Pr^{\max}_s(\lozenge G)$ and (ii) $u_k \in U_k$, where

$$U_k = \{u \in \mathbb{R} \mid \sigma_k \in \arg\max_{\sigma \in \mathfrak{S}^{\mathcal{M}}} \Pr^{\sigma}_s(\lozenge^{\leq k} G) + \Pr^{\sigma}_s(\square^{\leq k} S_?) \cdot u \text{ for all } s \in S_?\}.$$

Intuitively, the set $U_k \subseteq \mathbb{R}$ consists of all possible upper bounds u for which σ_k is still optimal. $U_k \subseteq$ is convex as it can be represented as a conjunction of inequalities with $U_0 = \mathbb{R}$ and $u \in U_k$ if and only if $u \in U_{k-1}$ and for all $s \in S_?$ with $\sigma_k(s) = \alpha$ and for all $\beta \in Act(s) \setminus \{\alpha\}$

$$\sum_{s' \in S} \mathbf{P}(s, \alpha, s') \cdot (x_{k-1}[s'] + y_{k-1}[s'] \cdot u) \geq \sum_{s' \in S} \mathbf{P}(s, \beta, s') \cdot (x_{k-1}[s'] + y_{k-1}[s'] \cdot u).$$

The algorithm maintains the so-called *decision value* d_k which corresponds to the minimum of U_k (or $-\infty$ if the minimum does not exist). Algorithm 4 outlines the

```
function $\mathit{findAction}(x, y, s, u)$
    if $u \neq \infty$ then
        return $\alpha \in \arg\max_{\alpha \in Act(s)} \sum_{s' \in S} \mathbf{P}(s, \alpha, s') \cdot (x[s'] + y[s'] \cdot u)$
    else
        return $\alpha \in \arg\max_{\alpha \in Act(s)} \sum_{s' \in S} \mathbf{P}(s, \alpha, s') \cdot (y[s'])$
```

Algorithm 3: Computation of optimal action.

```
function $\mathit{decisionValue}(x, y, s, \alpha)$
    $d \leftarrow -\infty$
    foreach $\beta \in Act(s) \setminus \{\alpha\}$ do
        $y_\Delta \leftarrow \sum_{s' \in S} (\mathbf{P}(s, \alpha, s') - \mathbf{P}(s, \beta, s')) \cdot y[s']$
        if $y_\Delta > 0$ then
            $x_\Delta \leftarrow \sum_{s' \in S} (\mathbf{P}(s, \beta, s') - \mathbf{P}(s, \alpha, s')) \cdot x[s']$
            $d \leftarrow \max(d, x_\Delta / y_\Delta)$
    return $d$
```

Algorithm 4: Computation of decision value.

procedure to obtain the decision value at a given state. Our algorithm ensures that u_k is only set to a value in $[d_k, u_{k-1}] \subseteq U_k$.

Lemma 4. *After executing Line 18 of Algorithm 2:* $u_k \geq \max_{s \in S_?} \mathrm{Pr}_s^{\max}(\Diamond G)$.

To show that u_k is a valid upper bound, let $s_{\max} \in \arg\max_{s \in S_?} \mathrm{Pr}_s^{\max}(\Diamond G)$ and $u^* = \mathrm{Pr}_{s_{\max}}^{\max}(\Diamond G)$. From Theorem 4, $u_{k-1} \geq u^*$, and $u_{k-1} \in U_k$ we get

$$\begin{aligned} u^* &\leq \max_{\sigma \in \mathfrak{S}^{\mathcal{M}}} \mathrm{Pr}_{s_{\max}}^{\sigma}(\Diamond^{\leq k} G) + \mathrm{Pr}_{s_{\max}}^{\sigma}(\Box^{\leq k} S_?) \cdot u_{k-1} \\ &= \mathrm{Pr}_{s_{\max}}^{\sigma_k}(\Diamond^{\leq k} G) + \mathrm{Pr}_{s_{\max}}^{\sigma_k}(\Box^{\leq k} S_?) \cdot u_{k-1} = x_k[s_{\max}] + y_k[s_{\max}] \cdot u_{k-1} \end{aligned}$$

which yields a new upper bound $x_k[s_{\max}] + y_k[s_{\max}] \cdot u_{k-1} \geq u^*$. We repeat this scheme as follows. Let $v_0 := u_{k-1}$ and for $i > 0$ let $v_i := x_k[s_{\max}] + y_k[s_{\max}] \cdot v_{i-1}$. We can show that $v_{i-1} \in U_k$ implies $v_i \geq u^*$. Assuming $y_k[s_{\max}] < 1$, the sequence $v_0, v_1, v_2, \ldots$ converges to $v_\infty := \lim_{i \to \infty} v_i = \frac{x_k[s_{\max}]}{1 - y_k[s_{\max}]}$. We distinguish three cases to show that $u_k = \min(u_{k-1}, \max(d_k, \max_{s \in S_?} \frac{x_k[s]}{1 - y_k[s]})) \geq u^*$.

- If $v_\infty > u_{k-1}$, then also $\max_{s \in S_?} \frac{x_k[s]}{1 - y_k[s]} > u_{k-1}$. Hence $u_k = u_{k-1} \geq u^*$.
- If $d_k \leq v_\infty \leq u_{k-1}$, we can show that $v_i \leq v_{i-1}$. It follows that for all $i > 0$, $v_{i-1} \in U_k$, implying $v_i \geq u^*$. Thus we get $u_k = \max_{s \in S_?} \frac{x_k[s]}{1 - y_k[s]} \geq v_\infty \geq u^*$.
- If $v_\infty < d_k$ then there is an $i \geq 0$ with $v_i \geq d_k$ and $u^* \leq v_{i+1} < d_k$. It follows that $u_k = d_k \geq u^*$.

Example 7. Reconsider the MDP $\mathcal{M}$ from Fig. 2(a) and goal states $G = \{s_3, s_4\}$. The maximal reachability probability is attained for a scheduler that always chooses β at state s_0, which results in $\Pr^{\max}(\lozenge G) = 0.5$. We now illustrate how Algorithm 2 approximates this value by sketching the first two iterations. For the first iteration *findAction* yields action α at s_0. We obtain:

$$x_1[s_0] = 0,\ x_1[s_1] = 0.1,\ x_1[s_2] = 0.1,\ y_1[s_0] = 0.8,\ y_1[s_1] = 0.9,\ y_1[s_2] = 0,$$
$$d_1 = 0.3/(0.8 - 0.4) = 0.75,\ \ell_1 = \min(0, 1, 0) = 0,\ u_1 = \max(0.75, 0, 1, 0) = 1.$$

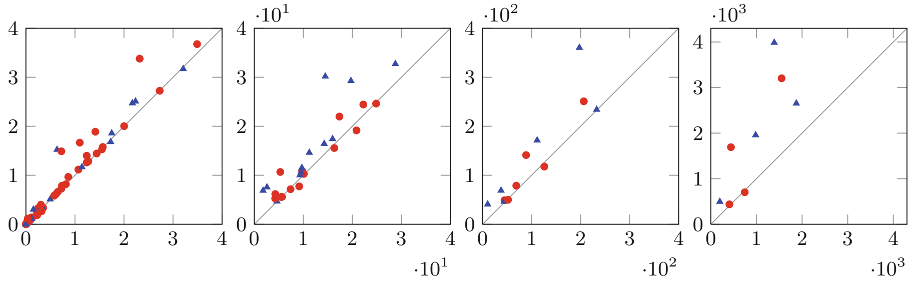
In the second iteration *findAction* yields again α for s_0 and we get:

$$x_2[s_0] = 0.08,\ x_2[s_1] = 0.19,\ x_2[s_2] = 0.1,\ y_2[s_0] = 0.72,\ y_2[s_1] = 0,\ y_2[s_2] = 0,$$
$$d_2 = 0.75,\ \ell_2 = \min(0.29, 0.19, 0.1) = 0.1,\ u_2 = \max(0.75, 0.29, 0.19, 0.1) = 0.75.$$

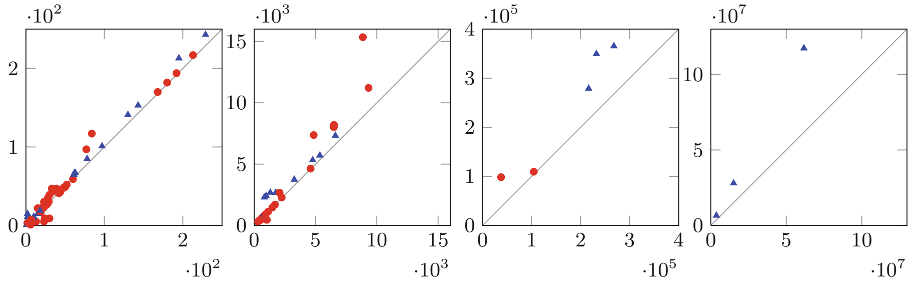
Due to the decision value we do not set the upper bound u_2 to $0.29 < \Pr^{\max}(\lozenge G)$.

Theorem 5. *Algorithm 2 terminates for any MDP $\mathcal{M}$, goal states G and precision $\varepsilon > 0$. The returned value r satisfies $|r - \Pr^{\max}(\lozenge G)| \leq \varepsilon$.*

The correctness of the algorithm follows from Theorem 4 and Lemma 3. Termination follows since $\mathcal{M}$ is contracting with respect to $S_0 \cup G$, implying $\lim_{k\to\infty} \Pr^{\sigma}(\square^{\leq k} S_?) = 0$. The optimizations for Algorithm 1 mentioned in Sect. 3.4 can be applied to Algorithm 2 as well.



(a) Model checking times (in seconds).



(b) Required iterations.

Fig. 3. Comparison of sound value iteration (x-axis) and interval iteration (y-axis).

5 Experimental Evaluation

Implementation. We implemented sound value iteration for MCs and MDPs into the model checker `Storm` [8]. The implementation computes reachability probabilities and expected rewards using explicit data structures such as sparse matrices and vectors. Moreover, Multi-objective model checking is supported, where we straightforwardly extend the value iteration-based approach of [22] to sound value iteration. We also implemented the optimizations given in Sect. 3.4.

The implementation is available at www.stormchecker.org.

Experimental Results. We considered a wide range of case studies including

– all MCs, MDPs, and CTMCs from the `PRISM` benchmark suite [23],
– several case studies from the `PRISM` website www.prismmodelchecker.org,
– Markov automata accompanying `IMCA` [24], and
– multi-objective MDPs considered in [22].

In total, 130 model and property instances were considered. For CTMCs and Markov automata we computed (untimed) reachability probabilities or expected rewards on the underlying MC and the underlying MDP, respectively. In all experiments the precision parameter was given by $\varepsilon = 10^{-6}$.

We compare sound value iteration (SVI) with interval iteration (II) as presented in [12,13]. We consider the Gauss-Seidel variant of the approaches and compute initial bounds ℓ_0 and u_0 as in [12]. For a better comparison we consider the implementation of II in `Storm`. [21] gives a comparison with the implementation of II in `PRISM`. The experiments were run on a single core (2GHz) of an HP BL685C G7 with 192GB of available memory. However, almost all experiments required less than 4GB. We measured model checking times and required iterations. All logfiles and considered benchmarks are available at [25].

Figure 3(a) depicts the model checking times for SVI (x-axis) and II (y-axis). For better readability, the benchmarks are divided into four plots with different scales. Triangles (▲) and circles (●) indicate MC and MDP benchmarks, respectively. Similarly, Fig. 3(b) shows the required iterations of the approaches. We observe that SVI converged faster and required fewer iterations for almost all MCs and MDPs. SVI performed particularly well on the challenging instances where many iterations are required. Similar observations were made when comparing the topological variants of SVI and II. Both approaches were still competitive if no a priori bounds are given to SVI. More details are given in [21].

Figure 4 indicates the model checking times of SVI and II as well as their topological variants. For reference, we also consider standard (unsound) value iteration (VI). The x-axis depicts the number of instances that have been solved by the corresponding approach within the time limit indicated on the y-axis. Hence, a point (x, y) means that for x instances the model checking time was less or equal than y. We observe that the topological variant of SVI yielded the best run times among all sound approaches and even competes with (unsound) VI.

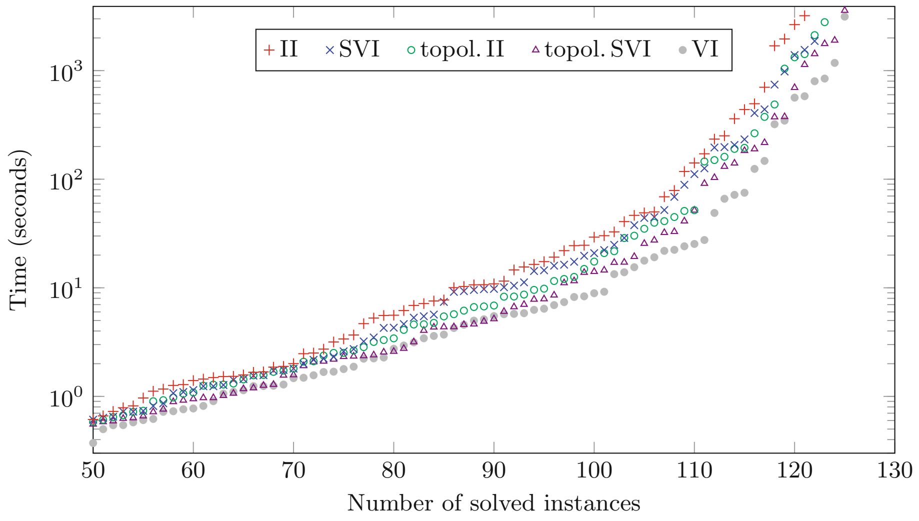


Fig. 4. Runtime comparison between different approaches.

6 Conclusion

In this paper we presented a sound variant of the value iteration algorithm which safely approximates reachability probabilities and expected rewards in MCs and MDPs. Experiments on a large set of benchmarks indicate that our approach is a reasonable alternative to the recently proposed interval iteration algorithm.

References

1. Puterman, M.L.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. Wiley, Hoboken (1994)
2. Feinberg, E.A., Shwartz, A.: Handbook of Markov Decision Processes: Methods and Applications. Kluwer, Dordrecht (2002)
3. Katoen, J.P.: The probabilistic model checking landscape. In: LICS, pp. 31–45. ACM (2016)
4. Baier, C.: Probabilistic model checking. In: Dependable Software Systems Engineering. NATO Science for Peace and Security Series - D: Information and Communication Security, vol. 45, pp. 1–23. IOS Press (2016)
5. Etessami, K.: Analysis of probabilistic processes and automata theory. In: Handbook of Automata Theory. European Mathematical Society (2016, to appear)
6. Baier, C., de Alfaro, L., Forejt, V., Kwiatkowska, M.: Probabilistic model checking. In: Clarke, E., Henzinger, T., Veith, H., Bloem, R. (eds.) Handbook of Model Checking, pp. 963–999. Springer, Cham (2018)
7. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: verification of probabilistic real-time systems. In: Gopalakrishnan, G., Qadeer, S. (eds.) CAV 2011. LNCS, vol. 6806, pp. 585–591. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22110-1_47

8. Dehnert, C., Junges, S., Katoen, J.-P., Volk, M.: A Storm is coming: a modern probabilistic model checker. In: Majumdar, R., Kunčak, V. (eds.) CAV 2017. LNCS, vol. 10427, pp. 592–600. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63390-9_31
9. Katoen, J., Zapreev, I.S.: Safe on-the-fly steady-state detection for time-bounded reachability. In: QEST, pp. 301–310. IEEE Computer Society (2006)
10. Malhotra, M.: A computationally efficient technique for transient analysis of repairable markovian systems. Perform. Eval. **24**(4), 311–331 (1996)
11. Daca, P., Henzinger, T.A., Kretínský, J., Petrov, T.: Faster statistical model checking for unbounded temporal properties. ACM Trans. Comput. Log. **18**(2), 12:1–12:25 (2017)
12. Baier, C., Klein, J., Leuschner, L., Parker, D., Wunderlich, S.: Ensuring the reliability of your model checker: interval iteration for markov decision processes. In: Majumdar, R., Kunčak, V. (eds.) CAV 2017. LNCS, vol. 10426, pp. 160–180. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63387-9_8
13. Haddad, S., Monmege, B.: Interval iteration algorithm for MDPs and IMDPs. Theor. Comput. Sci. **735**, 111–131 (2017)
14. Dai, P., Weld, D.S., Goldsmith, J.: Topological value iteration algorithms. J. Artif. Intell. Res. **42**, 181–209 (2011)
15. Baier, C., Katoen, J.P.: Principles of Model Checking. The MIT Press, Cambridge (2008)
16. Bertsekas, D.P., Tsitsiklis, J.N.: An analysis of stochastic shortest path problems. Math. Oper. Res. **16**(3), 580–595 (1991)
17. Giro, S.: Efficient computation of exact solutions for quantitative model checking. In: QAPL. EPTCS, vol. 85, pp. 17–32 (2012)
18. Bauer, M.S., Mathur, U., Chadha, R., Sistla, A.P., Viswanathan, M.: Exact quantitative probabilistic model checking through rational search. In: FMCAD, pp. 92–99. IEEE (2017)
19. Brázdil, T., et al.: Verification of Markov decision processes using learning algorithms. In: Cassez, F., Raskin, J.-F. (eds.) ATVA 2014. LNCS, vol. 8837, pp. 98–114. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11936-6_8
20. Haddad, S., Monmege, B.: Reachability in MDPs: refining convergence of value iteration. In: Ouaknine, J., Potapov, I., Worrell, J. (eds.) RP 2014. LNCS, vol. 8762, pp. 125–137. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11439-2_10
21. Quatmann, T., Katoen, J.P.: Sound value iteration. Technical report, CoRR abs/1804.05001 (2018)
22. Forejt, V., Kwiatkowska, M., Parker, D.: Pareto curves for probabilistic model checking. In: Chakraborty, S., Mukund, M. (eds.) ATVA 2012. LNCS, pp. 317–332. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33386-6_25
23. Kwiatkowska, M., Norman, G., Parker, D.: The PRISM benchmark suite. In: Proceedings of QEST, pp. 203–204. IEEE CS (2012)
24. Guck, D., Timmer, M., Hatefi, H., Ruijters, E., Stoelinga, M.: Modelling and analysis of markov reward automata. In: Cassez, F., Raskin, J.-F. (eds.) ATVA 2014. LNCS, vol. 8837, pp. 168–184. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11936-6_13
25. Quatmann, T., Katoen, J.P.: Experimental Results for Sound Value Iteration. figshare (2018). https://doi.org/10.6084/m9.figshare.6139052