Stopping Criteria for Value Iteration on Stochastic Games with Quantitative Objectives

Jan Křetínský[‡][†]
0000-0002-8122-2881

Tobias Meggendorfer[‡][*]
0000-0002-1712-2165

Maximilian Weininger[‡][*]
0000-0002-0163-2152

[*]*Institute of Science and Technology Austria*
Klosterneuburg, Austria

[†]*Masaryk University*
Brno, Czech Republic

[‡]*Technical University of Munich*
Garching bei München, Germany

***Abstract*—A classic solution technique for Markov decision processes (MDP) and stochastic games (SG) is value iteration (VI). Due to its good practical performance, this approximative approach is typically preferred over exact techniques, even though no practical bounds on the imprecision of the result could be given until recently. As a consequence, even the most used model checkers could return arbitrarily wrong results. Over the past decade, different works derived stopping criteria, indicating when the precision reaches the desired level, for various settings, in particular MDP with reachability, total reward, and mean payoff, and SG with reachability.**

In this paper, we provide the first stopping criteria for VI on SG with total reward and mean payoff, yielding the first anytime algorithms in these settings. To this end, we provide the solution in two flavours: First through a reduction to the MDP case and second directly on SG. The former is simpler and automatically utilizes any advances on MDP. The latter allows for more local computations, heading towards better practical efficiency.

Our solution unifies the previously mentioned approaches for MDP and SG and their underlying ideas. To achieve this, we isolate objective-specific subroutines as well as identify objective-independent concepts. These structural concepts, while surprisingly simple, form the very essence of the unified solution.

***Index Terms*—Stochastic games, value iteration**

I. Introduction

Markov decision processes (MDP) and stochastic games (SG) are the standard basic models for sequential decision making in the presence of uncertainty. While MDP can model either controllable behaviour or adversarial behaviour (such as schedulers or unknown environment), SG generalize MDP to encompass both at once. A *quantitative objective* for such a system is a random variable, assigning a real number to each run. A classic task is to find the optimal value v^* of its expectation, maximized (or minimized) over all strategies (a.k.a. policies, schedulers, or controllers) resolving the choices. For decades, this problem has been thoroughly investigated for both formalisms with respect to various quantitative objectives, such as discounted reward [1], total reward [2], long-run average reward (a.k.a. mean payoff) [3], [4], or reachability/safety [5] as an important special case of the latter two.

Value iteration (VI) [6] is a classic solution technique applicable to these settings [7], yielding a sequence of approximants converging in the limit to the actual optimal value v^*. It often is preferred over other solution methods due to its good practical performance, see [8] for a comparison on MDP. For the use in applications such as verification of safety-critical systems, guarantees on the *precision of the approximants* play a paramount role since the exact optimal value v^* may not be attained by any of the approximants within the given computation-time budget or in any finite time. Besides, it is typically sufficient to determine v^* with a specified precision. In this case an earlier termination of the algorithm is desirable, while still guaranteeing at least the required precision.

However, until recently, VI algorithms for non-discounted rewards provided no precision of the produced approximants and implementations in model checkers could yield arbitrarily wrong results [9]! Fortunately, recent works have equipped VI in several settings with *stopping criteria*, i.e. the algorithms produce not only approximants of v^* but also a bound on their imprecision, allowing VI to stop when the desired precision is reached. In other words, the algorithms produce guaranteed lower and upper bounds on v^*. Since these under- and over-approximations of v^* are steadily improving and accessible at any point of time during the (possibly infinite) run of the VI algorithm, VI becomes a so-called *anytime algorithm*, i.e. at every step in the algorithm, it can return the current estimate with its imprecision/error bound, and this bound converges to 0 in the limit. This has been achieved for MDP with various objectives such as reachability [10], [9], total reward until reaching the target [11], [12], [13], long-run average reward [14], but for SG due to their more complex structure only for reachability [15], [16] and multi-dimensional reachability [17]. As emphasized in the conclusion of [18], these shortcomings are still to be studied for SG.

Our contribution in this paper are the *first stopping criteria* for total reward and long-run average reward in SG. This enables a reliable use of VI, as required e.g. in model checkers. Moreover, our approach is a *unified solution* for the various objectives and systems. To better identify the commonalities of the various objectives and achieve deeper understanding, we provide the solution in two flavours. or two solutions?
The first algorithm reduces the problem on SG to a sequence of problems on MDP. These can be solved by any algorithm for MDP. Consequently, this approach directly profits from any improvements on solving MDP. Moreover, our algorithm generalizes to a stopping criterion for strategy iteration (SI) [19] in SG: While SI classically is required to be executed until the optimum is obtained in order to yield guarantees, we obtain

This research was funded in part by DFG projects 383882557 "SUV" and 427755713 "GOPro".

converging bounds and thus an anytime algorithm, too.

The second algorithm extends the guaranteed-precision VI from MDP to SG. It uses the bounded variant of VI [20], where VI iterates both from below and separately from above, thus providing its own error bounds. Since this algorithm operates *directly on SG*, it can localize the convergence difficulties (we even give a characterization of when this happens) and overcome them by local and thus possibly cheaper computation.

On the conceptual level, our work can be seen as an extension of previous work from *two perspectives*:

Firstly, more visibly in the first algorithm, we *extend the traditional VI* algorithms for the various objectives in SG to VI *with a stopping criterion*. Instead of computing one sequence converging to v^*, we compute two sequences monotonically converging from below and from above, forming the under- and over-approximations, respectively. Interestingly, depending on the objective, traditional VI algorithms may be reused to different extents. For instance, the traditional VI for reachability already yields a valid under-approximation and we only have to complement it with a valid over-approximation. Dually, for safety, the traditional VI provides an over-approximation to be complemented by an under-approximation. In contrast, for long-run average reward – a generalization of both reachability of safety – new approximations are needed on both sides since the original VI sequence is oscillating around the value v^*.

Secondly, more visibly in the second algorithm, we *extend the VI stopping criteria* for MDP and for reachability in SG *to SG with more complex objectives*. VI essentially computes a fixpoint of an update rule for the value in each states. The issue in getting both converging under- and over-approximation is that there may be more fixpoints because of cyclic dependencies of the values. In MDP, this takes the form of all states in a so-called *maximal end component* (MEC) having the same value; the solution is thus simply grouping all these states into a single one (often called MEC-quotient or collapsing MECs [10], [9]). While these MECs can be identified on the graph-theoretical level, in SG the solution requires a finer concept of so-called *simple end components* (SEC) [15], which depend on the values that we are in turn trying to compute (as explained in Example 2). This vicious cycle is broken by considering varying sets of states to be grouped only temporarily during the run of the algorithm, depending on the current approximations. Finally, in contrast to reachability, where remaining in such sets of states yields only values 0 or 1, yet another level of difficulty in grouping arises for total and average reward, where non-trivial values can be attained.

Our contribution can be thus summarized as follows:

- We give the first stopping criteria for VI on SG with total reward (in its three common variants [21]) and with long-run average reward, hence the first reliable VI algorithms. Moreover, they are in the form of anytime algorithms.
- Our approach provides a uniform framework for the new as well as previously studied cases with the standard quantitative objectives. In the course of this, we identify fundamental objective-independent concepts, such as "staying" and "exiting" values, together with the objective-specific ways to compute them, or a new definition of SEC. The new concepts apply uniformly to all considered objectives. Consequently, our solution is also more transparent. Finally, due to its generality, it yields a stopping criterion for strategy iteration, too.

A. Further Related Work

Beside VI there are several techniques used to solve MDP or SG. *Linear programming (LP)* typically yields solutions to MDP problems running in polynomial time. In contrast, for games, LP cannot capture the opposite goals of the players and we have to resort to quadratic programming [5], which, using current methods, is either practically infeasible or exponential in the size of the SG [22]. Besides, even in MDP the use of LP is practically limited, since it does not perform well on larger models [23], [8]. Thus, for both MDP and SG, usually either strategy iteration (SI) or value iteration are preferred.

Strategy iteration [19] computes a sequence of improving strategies in at most exponentially many iterations [24], [25]. Upon termination, the algorithm returns a guaranteed optimal strategy, i.e. one achieving the optimal value. However, the intermediate results only provide one-sided (converging) approximation of the result. (Indeed, SI improves and evaluates the strategies of one player only, since alternating improvement for both players may not terminate due to cycling [5].) Consequently, in order to obtain a guaranteed approximation, SI must finish the whole computation, which may be infeasible for large systems. Our stopping criterion extends to SI, providing a simple fix, and thus yields (to the best of our knowledge) the first anytime SI algorithm with guaranteed approximation even before termination.

Value iteration only is guaranteed to yield the precise result after a number of steps exponential in the size of the SG as well as the denominators of its quantities [7], which is infeasible even for toy games. Moreover, there are SG (even MDP) where this exponential number of steps indeed is necessary [26]. Despite the imprecision for the practically whole time of the algorithm execution, using VI to compute ε-precise solutions has been the most popular technique for MDP [27], [28] and SG [18] even without stopping criteria. This is because it (i) performs even better than SI on MDP [23], [8], (ii) uses heuristics exploiting that the result is imprecise anyway (asynchronous VI and partial exploration [20], [10], [29]), and (iii) performs on par with SI on SG [22].

The survey [7] provides a general treatment of VI on various models and objectives. While it discusses convergence and the worst-case analysis, we strengthen this with error bounds and anytime algorithms, providing better practical performance. It also conjectures VI is extensible to SG with long-run average reward, which we prove in this paper. For total reward, our starting point are algorithms presented in [21] which converge in the limit.

Finally, there are also further, mostly theoretical techniques to solve games, such as reducing SG with long-run average reward to SG with discounted reward or with reachability

[30] at the cost of introducing impractically small quantities, leading to a very poor convergence rate.

II. Preliminaries

In this section, we recall the basics of (turn-based) stochastic games, introduce relevant objectives and sketch existing solution approaches.

We write $\mathbb{R}$ and $\mathbb{R}_{\geq 0}$ to denote the real numbers and the non-negative real numbers, respectively. For a set S, $S^\star$ and S^ω refer to the set of finite and infinite sequences of elements of S, respectively. Whenever comparing two vectors x and y by $x \leq y$ or taking their $\max$ or $\min$, it is done point-wise. $\mathcal{D}(X)$ denotes the set of all *probability distributions* over a countable set X, i.e. mappings $d : X \to [0, 1]$ such that $\sum_{x \in X} d(x) = 1$. We also define $\max \emptyset = -\infty$ and $\min \emptyset = \infty$.

We refer to, e.g., [31], [32], [33], [34] for further information related to the topics discussed in the following.

A. Stochastic Systems

A *Markov chain (MC)* (e.g. [32]), is a tuple $\mathsf{M} = (\mathsf{S}, \delta)$, where S is a finite set of *states*, and $\delta : \mathsf{S} \to \mathcal{D}(\mathsf{S})$ is a *transition function* that for each state s yields a probability distribution over successor states.

A *(turn-based) stochastic game (SG)* (e.g. [5]) is a tuple $(\mathsf{S}_\square, \mathsf{S}_\bigcirc, \mathsf{A}, \delta)$, where $\mathsf{S}_\square$ and $\mathsf{S}_\bigcirc$ are finite, disjoint sets of states belonging to the *Maximizer* player $\square$ and the *Minimizer* player $\bigcirc$, respectively, and induce $\mathsf{S} := \mathsf{S}_\square \cup \mathsf{S}_\bigcirc$; further, A denotes a finite set of *actions* and we overload A to also act as a function assigning to each state s a set of non-empty *available actions* $\mathsf{A}(s)$; and $\delta : \mathsf{S} \times \mathsf{A} \to \mathcal{D}(\mathsf{S})$ is the *transition function* that for each state s and (available) action $a \in \mathsf{A}(s)$ yields a distribution over successor states.

A *Markov decision process (MDP)* (e.g. [31]) is an SG with only one player, i.e., $\mathsf{S}_\square = \emptyset$ or $\mathsf{S}_\bigcirc = \emptyset$.

For convenience, we write $\delta(s, s')$ and $\delta(s, a, s')$ instead of $\delta(s)(s')$ and $\delta(s, a)(s')$, respectively. Given a *state-action pair* $s \in \mathsf{S}$, $a \in \mathsf{A}(s)$, we use $\mathsf{Post}(s, a) := \{s' \in \mathsf{S} \mid \delta(s, a, s') > 0\}$ to denote the set of all successors of s under action a. For a state s, set $\preccurlyeq^s = \leq$ if $s \in \mathsf{S}_\square$ and $\geq$ otherwise. Analogously, opt^s is the $\max$-operator for $s \in \mathsf{S}_\square$ and the $\min$-operator otherwise, i.e. the preference of either player. We omit the superscript s when its clear from context. For a function $f : \mathsf{S} \to \mathbb{R}$, let $f(s, a) := \sum_{s' \in \mathsf{S}} \delta(s, a, s') \cdot f(s')$ denote the expected value of f achieved by following the action a once.

Semantics: We resolve choices by strategies, inducing a Markov chain with the respective probability space over infinite paths, as follows. Intuitively, a stochastic game is played in turns: In every state s, the player to whom it belongs chooses an action a from the set of available actions $\mathsf{A}(s)$ and the play advances to a successor state s' according to the probability distribution given by $\delta(s, a)$. Starting in a state s_0 and repeating this process indefinitely yields an infinite sequence $\rho = s_0 a_0 s_1 a_1 \cdots \in (\mathsf{S} \times \mathsf{A})^\omega$ such that for every $i \in \mathbb{N}_0$ we have $a_i \in \mathsf{A}(s_i)$ and $s_{i+1} \in \mathsf{Post}(s_i, a_i)$. We refer to such sequences as *(infinite) paths* or *plays* and use $\mathsf{Paths}_\mathsf{G}$ to denote the set of all such infinite paths in a given game G. Furthermore, we write ρ_i to denote the i-th state s_i in a path. *Finite paths* or *histories* are finite prefixes of a play, i.e. elements of $(\mathsf{S} \times \mathsf{A})^\star \times \mathsf{S}$ consistent with A and δ.

The path obtained by playing the game starting in a state s both depends on the choices of the two players as well as the random outcomes. The decision-making of the players is captured by the notion of *strategies*. In general, strategies are functions mapping a given history to a distribution over the actions available in the current state. However, in our case, *memoryless deterministic* strategies (abbreviated *MD strategies*), which choose a single action in each state irrespective of the history, are sufficient, as we argue later on. Thus we immediately restrict ourselves to this simpler case. Formally, a (MD) strategy of Maximizer $\sigma : \mathsf{S}_\square \to \mathsf{A}$ or Minimizer $\tau : \mathsf{S}_\bigcirc \to \mathsf{A}$ is a function mapping all states of the player to an available action, i.e. $\sigma(s) \in \mathsf{A}(s)$ for all s. A pair of strategies $\pi = (\sigma, \tau)$ is called *strategy profile*, and we define $\pi(s) := \sigma(s)$ if $s \in \mathsf{S}_\square$ and $\tau(s)$ otherwise.

After fixing one player's strategy, the outcome only depends on the choices of the other player and the randomness – this corresponds to an MDP. Analogously, once both players chose a strategy, the behaviour of the system is defined by a Markov chain. Given an appropriate pair of strategies (σ, τ) for a game G, we write $\mathsf{G}^{\sigma,\cdot}$ and $\mathsf{G}^{\cdot,\tau}$ for the MDP obtained after fixing either Maximizer's or Minimizer's strategy, respectively. Similarly, for a strategy profile $\pi = (\sigma, \tau)$, we write $\mathsf{G}^\pi = \mathsf{G}^{\sigma,\tau} = (\mathsf{S}, \hat{\delta})$, a Markov chain where $\hat{\delta}(s, s') := \delta(s, \pi(s), s')$.

Together with a state s, the Markov chain $\mathsf{G}^{\sigma,\tau}$ induces a unique probability distribution $\mathbb{P}^{\sigma,\tau}_{\mathsf{G},s}$ over the set of all infinite paths $\mathsf{Paths}_\mathsf{G}$ [32, Sec. 10.1] (where the set of paths starting in s has measure 1). For a random variable over paths $X : \mathsf{Paths}_\mathsf{G} \to \mathbb{R}$, we write $\mathbb{E}^{\sigma,\tau}_{\mathsf{G},s}[X]$ for the expected value of X under the probability measure $\mathbb{P}^{\sigma,\tau}_{\mathsf{G},s}$.

B. Objectives

Objectives formalize the "goal" of both players by assigning a value to each path. The two players are antagonistic, and, as their names suggest, Maximizer aims to maximize the obtained value, while Minimizer wants to minimize it (i.e. the game is zero-sum). We are interested in the *value of the game*, i.e. the optimal value the players can ensure. Formally, for an objective $\Phi : \mathsf{Paths}_\mathsf{G} \to (\mathbb{R} \cup \infty)$, the value of state s is defined as

$$\mathsf{V}_{\mathsf{G},\Phi}(s) := \sup\nolimits_\sigma \inf\nolimits_\tau \mathbb{E}^{\sigma,\tau}_{\mathsf{G},s}[\Phi] = \inf\nolimits_\tau \sup\nolimits_\sigma \mathbb{E}^{\sigma,\tau}_{\mathsf{G},s}[\Phi],$$

where the latter equality holds for all objectives we consider [35], [36], [37], [38], i.e. the games are *determined*. We consider four different objectives, namely reachability, safety, total reward, and mean payoff, which we now briefly introduce.

a) Reachability and Safety [5]: are specified by a set of target (respectively avoid) states $\mathsf{F} \subseteq \mathsf{S}$. We write $\Diamond\mathsf{F}$ to denote all paths ρ that reach F, i.e. there exists an i such that $\rho_i \in \mathsf{F}$. Then, for reachability we have that $\Phi(\rho) = 1$ if $\rho \in \Diamond\mathsf{F}$ and 0 otherwise. Dually, in safety, the goal is to avoid the given states, i.e. $\Phi(\rho) = 0$ if $\rho \in \Diamond\mathsf{F}$ and 1 otherwise. Note that maximizing a safety objective is equivalent to minimizing a reachability objective. We explicitly consider safety, since it allows us to illustrate some key concepts in a simpler setting. For both, MD strategies are sufficient [30].

b) Total Reward [21]: (also called expected reward) is specified by a *reward function* $\mathsf{r} : \mathsf{S} \to \mathbb{R}_{\geq 0}$. The value of the objective is the sum of the rewards that are accumulated along the run, formally $\Phi(\rho) = \sum_{i=0}^{\infty} \mathsf{r}(\rho_i)$. As rewards are non-negative, this sum always is defined, however it might be infinite. In any case, MD strategies are sufficient [21].

We remark that restricting to non-negative rewards is a standard assumption [21], [33]. In particular, once we allow for both positive and negative rewards, several issues arise, such as negative cycles, non-converging sums, etc., see [31, Sec. 5.2] for further remarks.

For the sake of readability, we only consider the most common variant of total reward for now. In Section VI, we discuss the (few) modifications required to apply our methods to the other two variants defined by [21].

c) Mean payoff (or long-run average reward) [3]: again is based on a reward function, but instead of the accumulated total reward, it considers the limit of the average reward. Formally, for a reward function $\mathsf{r} : \mathsf{S} \to \mathbb{R}_{\geq 0}$ the mean payoff of a path is defined by $\Phi(\rho) := \liminf_{n\to\infty}(\frac{1}{n}\sum_{j=0}^{n-1} \mathsf{r}(\rho_j))$. Negative rewards do not add additional complexity (see Appendix A-A), and we omit them for consistency. Again, MD strategies are sufficient [39, Thm. 1].

d) Canonical Forms: The objectives reachability and safety do not change their value after a target (or avoid) state is reached. Thus, without loss of generality, we assume that all states in F are *absorbing*, i.e. have only a single action which leads back to themselves. For total reward, we assume that all states have finite value: We can identify all states where the value is infinite through graph analysis [21, Sec. 4.3] and remove them from the game a-priori.

C. Value Iteration and Fixpoint Characterization

Value iteration (VI) is a classical and versatile solution approach applied in numerous settings. At its heart, VI relies, as the name suggests, on iteratively applying an operation to a value function, i.e. a mapping $f : \mathsf{S} \to \mathbb{R}$. In the following, we call such a function *assignment*, to avoid confusion with *the* value of a game $\mathsf{V}_{\mathsf{G},\Phi}$. The shape of both the assignment and the iterated update naturally depends on the problem at hand. Often, the update is derived from a fixpoint characterization of the assignment, which also is known as *Bellman optimality equations*. Since VI is central to this work, we briefly outline the structure of the classical value iteration solutions for our considered objectives. We refer to [7] for an in-depth discussion of value iteration and provide a brief overview.

For reachability, the values satisfy the equation [40]

$$\mathsf{V}(s) = \begin{cases} 1 & \text{if } s \in \mathsf{F}, \text{ and} \\ \mathsf{opt}_{a\in\mathsf{A}(s)} \sum_{s'\in\mathsf{S}} \delta(s,a,s') \cdot \mathsf{V}(s') & \text{otherwise,} \end{cases} \tag{1}$$

Classical value iteration starts with a vector x_0 which assigns 1 to all states in F and 0 to all others, and then iterates

$$x_{i+1}(s) = \mathsf{opt}_{a\in\mathsf{A}(s)} \sum_{s'\in\mathsf{S}} \delta(s,a,s') \cdot x_i(s'). \tag{2}$$

This update rule often is described as an instance of "Bellman backup" or "Bellman update". Iterating this rule converges to the correct value in the limit and furthermore is a monotonically improving lower bound on the true value, i.e. $x_i \leq x_{i+1} \leq \mathsf{V}_{\mathsf{G},\Phi}$ [40]. Further, x_i equals the optimal probability to reach the target in i steps. For the dual case, safety, we initially assign 0 to all avoid states F and 1 to all others and then iteratively apply Eq. (2), obtaining a sequence converging from above.

For total reward we additionally consider state rewards. The initial vector x_0 assigns 0 everywhere and we iterate

$$x_{i+1}(s) = \mathsf{r}(s) + \mathsf{opt}_{a\in\mathsf{A}(s)} \sum_{s'\in\mathsf{S}} \delta(s,a,s') \cdot x_i(s'). \tag{3}$$

This iteration computes the i-step optimal reward and converges to the true value in the limit.

For mean payoff, the classical value iteration does not compute a sequence converging to the mean payoff values. Instead, we compute the accumulated i-step reward x_i as in Eq. (3) and then estimate the mean payoff as x_i/i (or $x_i - x_{i-1}$). This approach was shown correct for both MDP [31, Sec. 9.4] and classical (non-stochastic) games [41, Sec. 2] and conjectured to be true for SG in [7, Sec. 5.2]. We provide a direct proof of this conjecture in Appendix A-C. See also Appendix A for more details on mean payoff in general.

We stress that there are two different notions, which however often (but not always) coincide, leading to potential confusion. First, there are the updates performed by value iteration ("Bellman updates"), e.g. the right hand side of Eq. (3). Second, there is the corresponding fixpoint characterization ("Bellman optimality equations"). These two notions agree for, e.g., reachability, but not for mean payoff. Both of these concepts will be relevant in this work, as well as a careful differentiation between them. As such, we refer to the value iteration operator of an objective Φ as Bellman updates and denote it by VI_Φ, whereas the right-hand side of the fixpoint characterization is referred to as fixpoint updates, written as Fix_Φ. Thus, $x_{i+1} = \mathrm{VI}_\Phi(x_i)$ and $\mathsf{V}_{\mathsf{G},\Phi} = \mathsf{Fix}_\Phi(\mathsf{V}_{\mathsf{G},\Phi})$. For example, for Φ as reachability, $\mathrm{VI}_\Phi = \mathsf{Fix}_\Phi$ is given by Eq. (2); in contrast, for Φ being mean payoff, VI_Φ is given by Eq. (3) while Fix_Φ is given by Eq. (2).

D. Approximate Solutions and Bounds

In this paper, we aim to derive an *ε-approximation* of the value. This means that for a game G, an objective Φ, and a precision ε, we want to compute an assignment x such that provably for all states s we have $|\mathsf{V}_\Phi(s) - x(s)| < \varepsilon$. This allows us to smoothly trade precision for computation time by adapting ε. Moreover, some practically efficient algorithms inherently are of approximative nature: For many applications, VI approaches typically yield solutions converging in the limit. However, convergence in the limit alone is not enough for our purposes, since we do not have any practical bounds on the error at any point in the iteration and thus cannot derive any ε-approximations. To tackle this issue, recent works, e.g. [10], [9], [15], introduced converging lower *and* upper bounds. This allows to stop the iteration once these values are sufficiently close to each other. Already in MDP, obtaining such bounds requires non-trivial reasoning related to so-called end components, defined in the following. One of our main contributions is to extend that reasoning, briefly summarized in the next section, to stochastic games.

E. Components and Exits

Intuitively, an end component is a set of states in which the system can remain forever, given suitable strategies. Formally, a pair (R, B), where $\emptyset \neq R \subseteq \mathsf{S}$ and $\emptyset \neq B \subseteq \bigcup_{s \in R} \mathsf{A}(s)$, is an *end component (EC)* [42] if (i) for all $s \in R$ and $a \in \mathsf{A}(s) \cap B$ we have $\mathsf{Post}(s, a) \subseteq R$, and (ii) for all $s, s' \in R$ there is a finite path $sa_0 \dots a_n s' \in (R \times B)^\star \times R$, i.e. the path stays inside R and only uses actions in B. Inclusion-maximal ECs are called *maximal end component (MEC)*.

We say $s \in R$ is an *exit* of R if there is an *exiting action* $a \in \mathsf{A}(s)$ with $\mathsf{Post}(s, a) \nsubseteq R$, and we write (s, a) exits R. We identify an EC with its set of states. Note that given two ECs (R_1, B_1) and (R_2, B_2) with $R_1 \cap R_2 \neq \emptyset$, their union $(R_1 \cup R_2, B_1 \cup B_2)$ also is an EC. Consequently, each state belongs to at most one MEC. The set of MECs can be determined in polynomial time [43]. Moreover, independent of the strategies, the play of an MDP / SG eventually remains inside a single MEC with probability one [42]. We can *restrict* a game G to an EC $E = (R, B)$, written $\mathsf{G}_{|E}$, by defining $\mathsf{S}' = R$, $\mathsf{A}' = B$, $\mathsf{A}'(s) = \mathsf{A}(s) \cap B$ and adapting δ accordingly. This captures all behaviours that can occur inside the EC E.

The corresponding notion on Markov chains are *bottom strongly connected components* (BSCCs), which are sets of states R that are (i) *strongly connected*, i.e. for every pair $s, s' \in R$ there is a non-empty finite path from s to s', (ii) *inclusion maximal*, i.e. there exists no strongly connected R' with $R \subsetneq R'$, and (iii) *bottom*, i.e. for all $s \in C, s' \in S \setminus C$ we have $\delta(s, s') = 0$.

III. Previous Approaches

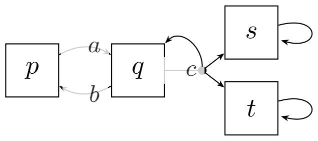
In this section, we illustrate the difficulties of obtaining converging bounds, as observed in recent works [10], [9], [15] and outline their key solution ideas on examples. In the next sections, we then explain how we unify these ideas and generalize them to stochastic games.

A. Non-converging Upper Bounds in MDP

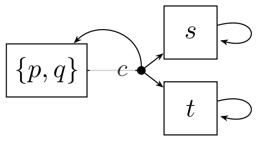
Recall that the iteration of Eq. (2) directly gives us correct lower bounds on the true value for any reachability objective. Thus, our goal is to find a similar kind of iteration sequence which gives *upper* bounds on the true value. Unfortunately, simply applying Eq. (2) to upper bounds does not work.

Example 1. Consider the example MDP in Fig. 1a together with the reachability objective $\mathsf{F} = \{t\}$. The iteration as given in Eq. (2) converges to the true value of $\frac{1}{2}$ in both states p and q, as illustrated in Fig. 1c. Note that it only converges in the limit, as within any finite number of steps, there is a positive chance to remain in q when using action c.

In order to obtain upper bounds, we may naively start with the greatest possible value of $\mathsf{U} = 1$ everywhere. However,



(a) The example MDP.



(b) The collapsed MDP.

(c) First few steps of value iteration applied to the example MDP.

i	$\mathsf{L}_i(p)$	$\mathsf{L}_i(q)$	$\mathsf{U}_i(p)$	$\mathsf{U}_i(q)$	$\mathsf{U}_i(\{p,q\})$
0	0	0	1	1	1
1	0	$1/3$	1	1	$2/3$
2	$1/3$	$4/9$	1	1	$5/9$

Figure 1: An example MDP inspired by [15] to showcase the problem of obtaining upper bounds for reachability (left) and its "collapsed" counterpart (right).

this also is a solution to Eq. (1), i.e. a (non-least) fixpoint of the Bellman operator, and thus the iteration remains at this incorrect value instead of converging to the true value. As an additional step, we can analyse the graph and identify state s as a state which cannot reach the target. However, even after setting its initial upper bound to 0, the iteration does not converge: States p and q keep their upper bound of 1 due to their cyclic dependency on each other's values. $\triangle$

We informally call such a situation a *spurious fixpoint*: Our iterates converge to some value which is not the correct value of the game, due to such cyclic dependencies. Identifying these dependencies is the key observation to obtain converging bounds. In [10], [9] the authors identify MECs as the root issue in MDP and propose to "collapse" them. Their fundamental insight is that all states in a MEC have the same value for reachability. Intuitively, once the system is in a MEC, it can ensure to eventually reach any state of this MEC with probability 1. Hence, we can move to the exit of the MEC obtaining the best value (later defined as *best exit value*) and continue from there. In other words, MECs form an equivalence class w.r.t. reachability. Collapsing essentially eliminates all internal behaviour of the MEC and replaces it with a single representative state comprising all exiting actions. This ensures that the Bellman iteration only chooses among all exits of the MEC, removing the internal cyclic dependency.

In our example, we collapse the MEC $(\{p, q\}, \{a, b\})$ to a single state $\{p, q\}$. The available actions of $\{p, q\}$ are those that exit the MEC, in this case only c, as shown in Fig. 1b. Now, value iteration also converges from above. Using variants of this observation, converging bounds have been obtained for all considered objectives on MDP [10], [9], [14], [12].

B. Collapsing in SG with Reachability

As outlined above, the key idea is to identify states whose values are in some sense "linked together". In MDP, we know that all states in a MEC have the same value, since the best exit can be reached independently of which state of the MEC the system is in. As we can identify MECs by graph analysis, we can simply collapse them and thus remove all cyclic

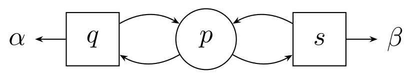


Figure 2: An example of a MEC where not all states have the same value, taken from [15]. Circles belong to Minimizer, rectangles to Maximizer. For clarity, the letters α and β represent the reachability values which can be obtained by taking the respective actions.

dependencies. However, analysing MECs is not sufficient for SG: The Minimizer may be able to prevent the Maximizer from reaching the best exit and vice versa.

Example 2. Consider the MEC in Fig. 2. First, assume that $\alpha > \beta$. Starting from state s, Maximizer cannot reach the best exit to achieve α, since in state p Minimizer can choose the action leading back to s. Thus, p and s have a value of β, while q has the value of α. If $\beta > \alpha$, we analogously get that p and q obtain a value of α, while s has value β. As such, the corresponding notion of "collapsible" states is not only graph-theoretic, but also depends on the numeric values. △

In an effort to distil the essence of MEC collapsing for stochastic games, [15] introduces the notion of *simple end components (SECs)*. Intuitively, an EC E is simple if all states in E have the same value, i.e. for all $s, s' \in E$ we have that $\mathsf{V}_{\mathsf{G},\Phi}(s) = \mathsf{V}_{\mathsf{G},\Phi}(s')$. As mentioned, in the MDP case each EC is simple. Unfortunately, this is not the case in SGs, and SECs seem to be much more elusive than MECs, as we illustrate with Example 2. In particular, SECs do not only depend on the graph structure but also on the concrete values of each state. However, this is what we are trying to compute in the first place! Additionally, since we are approximating, we may *never* be certain which states form a SEC. There can be several SECs in one MEC, thus their values depend on each other and there is no topological ordering in which we could evaluate the values which we rely on (see Fig. 2). Thus we cannot directly extend the idea of collapsing from MDP, as we might "commit" to a wrong guess of which states belong together.

As an alternative approach, [15] propose "deflating". We only summarize it very briefly and provide further, related insight in the following sections, where we present our more general approaches based on this idea. In essence, the algorithm "guesses" which states might form a SEC and then conservatively lowers the upper bound of all states in this presumed SEC. For an intuitive view, deflating can be seen as collapsing an EC, performing an update on the "collapsed" representative, and then "un-collapsing" the EC again. As such, we do not commit to considering a particular EC as simple forever. It turns out that a meticulous choice of SEC-candidates is sufficient to obtain convergence.

IV. Global: Reducing to MDP

We present our first algorithm for computing convergent under- and over-approximation of the value in SG in an objective-independent way. This "global approach" elegantly delegates reasoning to existing algorithms for MDP. The next section lifts that same idea in a more local way to the world of SG, pinpointing the root of the convergence issues.

Algorithm 1 Generic bounded value iteration based on global reduction to MDP

```
Input: SG $\mathsf{G}$, Objective $\Phi$, initial state $s_0$, and precision $\varepsilon$
Output: $(\mathsf{L}, \mathsf{U})$ such that $\mathsf{L} \leq \mathsf{V} \leq \mathsf{U}$ and $\mathsf{U}(s_0) - \mathsf{L}(s_0) \leq \varepsilon$
 1: $x_0 \leftarrow \textsc{InitVI}_\Phi$                      # Classic VI for $\Phi$
 2: $\mathsf{L}_0(\cdot) \leftarrow 0$, $\mathsf{U}_0(\cdot) \leftarrow \infty$                # Initialize bounds
 3: $i \leftarrow 0$
 4: repeat
 5:     $i \leftarrow i + 1$
        # Recommender procedure
 6:     $x_i \leftarrow \mathsf{VI}_\Phi(x_{i-1})$                 # Classic VI for $\Phi$
 7:     $\sigma, \tau \leftarrow$ strategies inferred from $x_i$ (see Lemma 1)
        # Compute bounds (pointwise maximum / minimum)
 8:     $\mathsf{L}_i \leftarrow \max(\mathsf{L}_{i-1}, \mathsf{V}(\mathsf{G}^{\sigma,\cdot}))$
 9:     $\mathsf{U}_i \leftarrow \min(\mathsf{U}_{i-1}, \mathsf{V}(\mathsf{G}^{\cdot,\tau}))$
10: until $\mathsf{U}_i(s_0) - \mathsf{L}_i(s_0) \leq \varepsilon$
```

The (surprisingly simple) key insight is as follows. Consider a game G and an arbitrary strategy τ of Minimizer. We fix this strategy and apply it in the SG to obtain an MDP $\mathsf{G}^{\cdot,\tau}$. If τ is optimal, then playing optimally in $\mathsf{G}^{\cdot,\tau}$ exactly gives the value of the game. Otherwise, if τ is sub-optimal, then Maximizer can exploit Minimizer's mistakes by answering optimally, possibly even obtaining a higher value, *but certainly not lower*. Thus, in both cases, fixing Minimizer's strategy τ and determining the value Maximizer can achieve against it, i.e. solving the MDP $\mathsf{G}^{\cdot,\tau}$, yields a *correct upper bound* on the actual value. Formally, for a strategy τ', we get

$$\mathsf{V}_{\mathsf{G},\Phi} = \inf_\tau \sup_\sigma \mathbb{E}^{\sigma,\tau}_{\mathsf{G},s}[\Phi] \leq \sup_\sigma \mathbb{E}^{\sigma,\tau'}_{\mathsf{G},s}[\Phi] = \mathsf{V}_{\mathsf{G}^{\cdot,\tau'},\Phi} \quad (4)$$

Dually, fixing strategies σ of Maximizer yields *lower* bounds:

$$\mathsf{V}_{\mathsf{G}^{\sigma,\cdot},\Phi} \leq \mathsf{V}_{\mathsf{G},\Phi}. \quad (5)$$

This immediately gives rise to our first algorithm, mixing this insight together with classical VI to obtain converging bounds.

A. Algorithm

We first present the algorithm template and discuss technical details later on. As mentioned, Algorithm 1 is based on classic VI without bounds, initialized and performed on Lines 1 and 6, respectively. We use $\textsc{InitVI}_\Phi$ to denote the initialization as described in Section II-C (e.g. for reachability 1 if $s \in \mathsf{F}$ and 0 otherwise). Additionally, we initialize and improve lower and upper bounds on the value on Lines 2, 8 and 9, respectively, until sufficient precision is detected (Line 10).

Now, for *soundness* of the algorithm, i.e. validity of the bounds, consider Eqs. (4) and (5). Lines 8–9 update the bounds this way, depending on the strategies "recommended" in Line 7. No matter which strategies are recommended, the update is correct. To ensure *convergence*, it is sufficient that at some point, optimal strategies of Maximizer and Minimizer are chosen. Indeed, if the chosen strategy σ or τ is optimal, then the values on the induced MDP equal the value of the game. We discuss how we can efficiently derive strategies from the iterates x_i in Lemma 1.

We emphasize that this observation is *independent* of the objective Φ and provides a surprisingly direct way to delegate the computation of bounds for an objective to solution methods for MDP. In a sense, identifying "problematic ECs" is done by eventually having optimal strategies inferred from x_i and dealing with them is delegated to MDP reasoning.

B. Correctness and Termination

According to the intuitive argumentation above, for the proof of correctness, we do not require Lines 1, 6 and 7 to exactly instantiate the classical value iteration. Instead, we use the more general assumption of a *recommender procedure*, which yields a sequence of strategies and eventually converges to the correct strategies. Afterwards we show that value iteration satisfies this assumption, but also give other suggestions for the recommender procedure.

Definition 1. A *(strategy) recommender* is a sequence (σ_i, τ_i) of pairs of MD strategies such that σ_i is optimal for some i and τ_j optimal for some j. A recommender is *stable* if there exists an index k such that σ_j and τ_j are optimal for all $j \geq k$.

Note that σ_j and τ_j do not need to remain constant even for a stable recommender, they only eventually all need to be optimal. This definition together with the previous considerations immediately yield both correctness and termination:

Theorem 1. *If the sequence (σ_i, τ_i) produced on Line 7 is a strategy recommender, then Algorithm 1 is correct and terminates, i.e. for all i, we have $\mathsf{L}_i \leq \mathsf{V}_{\mathsf{G},\Phi} \leq \mathsf{U}_i$, and for every $\varepsilon \geq 0$, there is i with $\mathsf{U}_i(s_0) - \mathsf{L}_i(s_0) \leq \varepsilon$.*

Proof: Correctness: Follows from Eqs. (4) and (5).
Termination: By definition of strategy recommender, the algorithm eventually evaluates the MDP induced by the optimal strategies. This directly gives us the true value of the game for the lower and upper bound, respectively, due to the optimality of MD strategies. ■

Note that once optimal strategies are recommended, we obtain the exact value and may converge faster than x_i.

Remark 1. We require a strategy recommender to yield memoryless deterministic strategies mainly for technical simplicity. In particular, memorylessness implies that (i) states and end components of the induced MDP have a direct correspondence to their counterparts in the game, i.e. every EC in the induced MDP is also an EC in the SG, and (ii) that only finitely many distinct recommendations exists, simplifying several arguments. This is not a restriction for our considered objectives, since MD strategies are sufficient for all of them and both VI as well as SI yield such strategies by default.

In order to obtain a strategy recommender, we formalize a folklore result for value iteration, allowing us to obtain strategies from the iterates.

Lemma 1. *Fix an initial pair of strategies (σ_0, τ_0). For any Maximizer state s, define $A_i(s)$ as the set of actions in state s that witness the update of the iterated vector x_i, i.e. all actions in $\arg\operatorname{opt}_{a \in \mathsf{A}(s)} x_i(s,a)$. If $\sigma_{i-1}(s) \in A_i(s)$, define $\sigma_i(s) = \sigma_{i-1}(s)$. Otherwise, pick any $a \in A_i(s)$ and set $\sigma_i(s) = a$. We analogously define τ_i. Then, the resulting sequence (σ_i, τ_i) is a stable strategy recommender for all considered objectives.*

Proof sketch: Intuitively, a strategy that performs optimally for a very long time actually behaves like the infinite horizon optimal ones for our objectives. However, the detailed reasoning is surprisingly intricate (and objective-specific). In particular, keeping the previous choice if possible is important, picking any optimal action may be incorrect. We provide a proof together with an illustrative example in Appendix C-A. We highlight the auxiliary Lemma 10, not included due to space constraints, which we believe to be useful for other works dealing with value iteration. ■

We remark that instead of keeping the previous choice if possible, one can similarly choose a strategy randomizing uniformly over $A_i(s)$ (also discussed in Appendix C-A).

The desired results for our algorithm follows immediately.

Theorem 2. *Algorithm 1 (as displayed with classical VI) is correct and terminates.*

Note that Algorithm 1 does not modify the intermediate computations of classical value iteration, but instead only complements them with computing lower and upper bounds (we discuss how the VI iterates can be incorporated into the bound computation for some objectives in Section VI). Solving the resulting MDPs takes polynomial time by classical encoding into linear programs [31]. Consequently, any results on the speed of convergence immediately carry over from classical value iteration, i.e. it can take exponential time in the worst case [26] and never more [7].

Remark 2. The assumptions of a recommender procedure are also satisfied by other procedures, for example, even by the naive enumeration of all strategies. More interestingly, it is also satisfied by classical strategy iteration, see [19], [44], [22]. In a nutshell, SI fixes a Maximizer strategy σ_i in each iteration and computes the best response of the opponent τ_i. Then, σ_{i+1} is obtained by picking optimal actions against τ_i. This eventually converges to an optimal pair of strategies [5].[(1)] Thus, (σ_i, τ_i) satisfies the conditions of a strategy recommender and Algorithm 1 can also be used to complement strategy iteration with convergent anytime bounds. See Appendix D for pseudocode of this approach and further discussion.

V. Local: Remaining in the World of SG

Our first algorithm addresses the convergence issues of VI by globally applying MDP reasoning, soundly under- and over-approximating the value in every iteration. In contrast, our second algorithm, inspired by [15], avoids repeatedly solving

[(1)]Interestingly, improving the strategies alternatingly for both players may not converge to optimal strategies and does not yield a recommender [5].

full MDPs and instead applies specialized reasoning only locally. This local view reveals important objective-independent concepts that allow us to pinpoint exactly where and when a problem for VI arises. Before the technical discussion, we provide an intuitive overview and exemplify the concepts.

a) Why does VI not converge?: Spurious Fixpoints. In Section III, we explained that cyclic dependencies of the iterates x_i prevent VI from converging, which we informally introduced as spurious fixpoints. We discuss another example to further illustrate how these fixpoints arise and why they pose a problem for VI. Consider the (fragment of an) SG in Fig. 3 with the EC $E = \{p, q\}$ and a reachability objective. Suppose that the true values outside of the EC are as given, i.e. 0.1 and 0.8 respectively, and the iterates x_i of value iteration have converged to these values outside of the EC. The Bellman (and fixpoint) updates are $x_{i+1}(p) = \max\{0.1, x_i(q)\}$ and $x_{i+1}(q) = \min\{0.8, x_i(p)\}$. Clearly, any value $0.1 \leq v \leq 0.8$ assigned to p and q, i.e. $x_i(p) = x_i(q) = v$, yields a fixpoint thereof: Given such v, both players prefer to "remain inside" the EC, relying on each others values – a cyclic dependency – and the iterates x_i do not converge to the correct value. Note that in particular the case of $v = 0.8$ arises for reachability upper bounds computed by VI.

b) What causes spurious fixpoints?: Simple End Components. Intuitively, an EC is simple if both players think that they neither gain nor lose anything by remaining in the EC for an arbitrary number of steps. (We formally define this notion in Definition 3.) If moreover the exiting actions of the EC do not seem better to both players, they will prefer remaining inside the EC. Concretely, in our example from Fig. 3, going from state p to q does not reduce Maximizer's chances to reach the goal: Minimizer can only exit and provide a higher value (0.8) or return back to p. Dually, Minimizer is content with moving the play back to q for the same reason. Since staying for any finite number of steps is optimal, VI (applied to upper bounds) cannot determine that Maximizer actually has to leave in order to have any chance to reach the goal, as it only computes a finite-horizon probability. Dually, for a safety objective, Minimizer does not realize that staying forever actually provides a value of 1 and leaving in q is optimal. We prove in Lemma 3 that indeed SECs are the *only reason* for spurious fixpoints to exist.

To further validate this intuition, assume the roles of Maximizer and Minimizer are exchanged in Fig. 3 and again suppose that $x_i(p) = x_i(q) = v$ for some $0.1 \leq v \leq 0.8$. Here, no problem arises: Minimizer immediately uses the exit of state p and obtains 0.1, while Maximizer uses the exit of q to obtain the higher value of 0.8 – the only fixpoint is given by the true values. Observe that the EC is not simple: Neither player wants to stay; leaving is strictly preferable for both.

c) How to handle SECs?: Staying and exit values. As pointed out, VI fails to handle exactly those situations where remaining inside an EC arbitrarily long is an optimal choice. Thus, we need to devise a mechanism to inform both players about what happens by remaining inside the EC *forever*. This is formalized as *staying value* (see Definition 4). Intuitively,

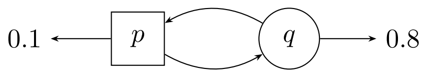


Figure 3: Example fragment of an SG to illustrate fixpoint issues. We omit action labels for clarity.

each player can at most achieve the staying value or the best exiting value (we formalize this intuition in Lemma 5). For example, in Fig. 3, Maximizer cannot obtain more than the maximum of his best exit (0.1) and the staying value (0). Thus, we can safely *deflate* $x_i(p)$ to 0.1, resolving any spurious fixpoint. Dually, if the objective were safety and we compute lower bounds, we observe that Minimizer cannot reduce the value below the best exit (0.8) and the staying value (1), leading us to *inflate* $x_i(q)$ to 0.8. As we show later, this treatment is both sound and sufficient to obtain convergence.

d) Summary: We identify SECs as the central cause for non-convergence of value iteration and provide way of treating them, using staying and exit values. The key insight is that if players are content with remaining in an EC for arbitrarily time, we need to inform them about what happens "at infinity".

We highlight that our concepts allow us to simplify, unify, and extend findings of the last decade, from MEC collapsing [10], [9] to deflating [15] (see Section VI). Moreover, these concepts are agnostic of the objective, only requiring it to be of a very general form, which we introduce as *fixpoint-linear*.

A. Fixpoint-Linear Objectives

To start our technical discussion, we first introduce a unified view of our considered objectives. Recall that a central aspect (and root cause for spurious fixpoints) is the fixpoint characterization of each objective. By considering the shape thereof, we classify a broad spectrum of *fixpoint-linear* objectives.

Each of the discussed objectives Φ has an associated *fixpoint characterization*, i.e. a function Fix_Φ which given an assignment f and state s yields the supposed value for this state based on its successors. Concretely, we have $\mathsf{Fix}_\Phi(f, s) := \mathsf{opt}_{a \in \mathsf{A}(s)} f(s, a)$ for reachability, safety, and mean payoff[(2)], and $\mathsf{Fix}_\Phi(f, s) := \mathsf{opt}_{a \in \mathsf{A}(s)} \mathsf{r}(s) + f(s, a)$ for total reward. Note that these are all of quite similar, linear shape.

Definition 2. An objective Φ is called fixpoint-linear if for every game G (in canonical form), the value is a solution to an (affine) linear equality of the form

$$\begin{aligned} f(s) &= \mathsf{opt}_{a \in \mathsf{A}(s)} \mathsf{off}_\Phi(s) + f(s, a) \\ &= \mathsf{opt}_{a \in \mathsf{A}(s)} \mathsf{off}_\Phi(s) + \sum\nolimits_{s' \in \mathsf{S}} \delta(s, a, s') \cdot f(s') \end{aligned}$$

for some $\mathsf{off}_\Phi(s) \geq 0$ in every state s.[(3)] We call the right hand side *fixpoint update of* Φ, written $\mathsf{Fix}_\Phi(f, s)$ for a function f.

Note that $\mathsf{off}_\Phi(s) = \mathsf{r}(s)$ for total reward and $\mathsf{off}_\Phi \equiv 0$ for all others. Many classical objectives are of this form, and even

(2)Recall that classical value iteration for mean payoff uses a different update, as explained in Section II-C.

(3)We deliberately exclude discounting objectives in this definition due to technical reasons and discuss how they can be handled in Section VI.

more can be reduced to this; see Section VI for discussion. However, this certainly is not true for all objectives, such as the (non-linear) conditional value-at-risk [45], [46].

B. Which End Components are Simple?

With these notions at hand, we introduce our generalization of simple end components. Throughout this section, we fix a game G and fixpoint-linear objective Φ.

Our intuitive overview identifies two conditions which jointly allow spurious fixpoints to arise: we may run into problems if (i) inside an EC an assignment f is a fixpoint and (ii) the actions of the EC are preferred over exiting actions by both players. As we confirm later, f might be a spurious fixpoint exactly in this case. We begin by showing that for fixpoint-linear objectives, end components which exhibit such a "fixpoint behaviour" already have a very specific shape:

Lemma 2. *Let f an assignment and $E = (R, B)$ an EC. Then*

$$f(s) = \mathsf{off}_\Phi(s) + f(s, a) \text{ for all } s \in R, a \in \mathsf{A}(s) \cap B,$$

implies $\mathsf{off}_\Phi(s) = 0$ for all $s \in R$.

See Appendix C-B for a formal proof.

The first condition means that f satisfies the fixpoint equation for Φ on $\mathsf{G}_{|E}$, i.e. the game restricted to E. With our intuition in mind, this means that only ECs where $\mathsf{off}_\Phi(s) = 0$ could be a problem. This motivates the following definition.

Definition 3. Let $E = (R, B)$ an EC and f an assignment. We call E a *simple end component for f* (SEC for f) if

$$f(s) = f(s, a) \text{ for all } s \in R, a \in \mathsf{A}(s) \cap B.$$

Inclusion maximal SECs are called *maximal simple end component* (MSEC). We write "SEC" instead of "SEC *for* $\mathsf{V}_{\mathsf{G},\Phi}$".

In [15], SECs are introduced as ECs where all states have the same value, which for reachability implied that the EC potentially allows for spurious fixpoints. It turns out that our definition of SECs implies that view.

Corollary 1. *Let f an assignment and E an EC. Then, E is a SEC for f if and only if (i) $f(s) = f(s')$ and (ii) $\mathsf{off}_\Phi(s) = 0$ for all $s, s' \in E$.*

See Appendix C-C for a formal proof. This motivates naming such ECs "simple": All states have the same value and no rewards are gained; arguably as simple as an EC can be.

Matching the discussed intuition, SECs indeed exactly pinpoint spurious fixpoints:

Lemma 3. *Let f a fixpoint for Fix_Φ. If $f \not\equiv \mathsf{V}_{\mathsf{G},\Phi}$, then there exists a SEC E for f where $f(s) \neq \mathsf{V}_{\mathsf{G},\Phi}(s)$ for all $s \in E$.*

The proof can be found in Appendix C-D.

Recall that spurious fixpoints are the sole reason why fixpoint updates do not converge. Additionally, Lemma 3 shows that the only reason why a spurious fixpoint f can exist is due to SECs: If f is fixpoint for Fix_Φ, i.e. applying fixpoint updates does not change f, then $f \equiv \mathsf{V}$ if and only

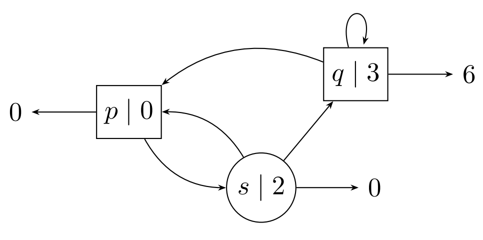


Figure 4: Example stochastic game to illustrate the definition of staying and exit value. We write the reward assigned to each state next to its name.

if $f(s) = \mathsf{V}(s)$ on all SECs for f. Thus, we identified the root cause for spurious fixpoints. However, we do not yet have a tool to deal with them. As discussed, SECs are exactly those places where both players think they can delay the play indefinitely and we need to inform them what happens if they indeed stay forever.

C. Should I Stay or Should I Go?

In order to distinguish the case of (eventually) leaving and staying forever, we introduce the notions of staying value and best exit, and treat them through *deflating* and *inflating*. In a sense, these correspond to offering a glimpse at what could happen in the infinite horizon.

Definition 4. Fix an EC $E = (R, B)$. The staying value of a state $s \in E$ is defined as

$$\mathsf{stay}_{\mathsf{G},\Phi}(E, s) := \mathsf{V}_{\mathsf{G}_{|E},\Phi}(s),$$

where $\mathsf{G}_{|E}$ is the SG G restricted to states and actions in E.

The *best exit value* (from E under an assignment f) of Maximizer and Minimizer are defined as

$$\mathsf{exit}^{\Box}_f(E) := \max_{s \in R \cap \mathsf{S}_\Box, (s,a) \text{ exits } E} f(s, a) \quad \text{and}$$
$$\mathsf{exit}^{\bigcirc}_f(E) := \min_{s \in R \cap \mathsf{S}_\bigcirc, (s,a) \text{ exits } E} f(s, a).$$

Note that we defined $\max \emptyset = -\infty$ and $\min \emptyset = \infty$, i.e. the "best exit" of Maximizer in an EC without exits is $-\infty$.

Example 3. Consider the EC $E = \{p, q, s\}$ of the SG from Fig. 4 together with a mean payoff objective. States p and s have a staying value of 1 in E: When restricted to the MEC, Minimizer always moves back to p instead of going to q. State q has a staying value of 3 by virtue of its self-loop. The best exit of Maximizer is given by 6 in state q, which is also what Maximizer can realize in q. State p has a value of 0 (which is strictly less than both staying value and Maximizer's best exit), since the Minimizer chooses to exit the EC in state s, using Minimizer's best exit. △

Recall our intuition that issues can arise exactly when both players prefer staying over leaving under the current values. Given our definitions, we can now formalize this claim.

Lemma 4. *Let f an assignment and E a SEC for f. We have $\mathsf{exit}^{\Box}_f(E) \leq f(s) \leq \mathsf{exit}^{\bigcirc}_f(E)$ for any $s \in E$* if and only if *a fixpoint update does not modify the value of f on the EC.*

Proof: Since E is SEC, by Lemma 2, we have that $f(s) = f(s') = c$ for all $s \in E$.

Observe that the backward direction follows immediately: If f is not modified on the EC it means that in every state the value c was weakly preferred over all exiting options.

For the forward direction, fix any Maximizer state s. Since $\mathsf{exit}_f^{\Box}(E) \leq c$, we also have that $f(s,a) \leq f(s,b)$ for any (s,a) exiting E and (s,b) remaining in E. This dually holds for Minimizer states. Consequently, the fixpoint update always considers the interior actions, yielding $\mathsf{Fix}_\Phi(f,s) = c = f(s)$ for every $s \in E$. ■

We apply this lemma to our running example.

Example 4. Recall the example of Fig. 3. Then, we have $\mathsf{exit}_{x_i}^{\Box}(E) = 0.1 < 0.8 = \mathsf{exit}_{x_i}^{\bigcirc}(E)$. As discussed, any value $0.1 \leq v \leq 0.8$ assigned to p and q yields a fixpoint of Fix_Φ (which is exactly what Lemma 4 yields). △

So, if both players believe they prefer staying based on the current assignment, we may run into trouble. To resolve this, we want to inform them about the consequences of staying. We show that in a SEC, the value of any state is bounded from above by the maximum of its staying value and the best exit value of Maximizer. Naturally, the dual statement holds for the Minimizer.

Lemma 5. *Fix a memoryless Minimizer strategy τ. Let $E = (R,B)$ be a SEC and f an upper bound on the value $\mathsf{V}_{\mathsf{G},\Phi}$. Then, we have for every $s \in R$*

$$\mathsf{V}_{\mathsf{G},\Phi}(s) \leq \max(\mathsf{stay}_{\mathsf{G}^{\cdot,\tau},\Phi}(E,s), \mathsf{exit}_f^{\Box}(E)). \tag{6}$$

Dually, for a Maximizer strategy σ and f lower bound, we get

$$\mathsf{V}_{\mathsf{G},\Phi}(s) \geq \min(\mathsf{stay}_{\mathsf{G}^{\sigma,\cdot},\Phi}(E,s), \mathsf{exit}_f^{\bigcirc}(E)). \tag{7}$$

The (rather technical but objective-independent) proof can be found in Appendix C-E. Note that the actual value may even be strictly smaller than both staying value and best exit, for example when Minimizer can prevent reaching the best exit, see Example 3. We call applying Eq. (6) *deflating* (reducing potentially "blown up" values) and dually Eq. (7) *inflating*.

With this idea at hand, we derive further insights for our considered objectives.

Remark 3. For reachability, observe that the staying value in all non-target SECs is 0 and iterates provide a converging lower bound. Thus, Eq. (6) yields $\mathsf{V}_{\mathsf{G},\Phi}(s) \leq \mathsf{exit}_{x_i}^{\Box}(E)$: The best value Maximizer can achieve is at most the best value achievable by leaving the EC. To visualize this further, consider Fig. 3 again. We see that $\mathsf{exit}_{x_i}^{\Box}(E) = 0.1$ which indeed is an upper bound to the correct value: Maximizer may think taking the staying action is not harmful – either Minimizer "gives" 0.8 or the play returns back to state p. However, by revealing the consequences of doing this forever (the actual staying value), Maximizer realizes that instead the best exit is the best possible chance in this EC, reducing the upper bound to 0.1. Moreover, applying this update in state p is enough to resolve convergence problems. This holds in general: Since for reachability iterates converge to the correct value from below on their own, we never need to inflate them.

In the case of safety, we dually observe that the staying value is 1 and iterates converge from above, meaning that only inflation is necessary. This contrasts mean payoff, where the staying value could actually lie between the best exits and we need to consider both sides.

Algorithm 2 Bounded value iteration with local reasoning

```
Input: SG $\mathsf{G}$, Objective $\Phi$, initial state $s_0$, and precision $\varepsilon$
Output: $(\mathsf{L},\mathsf{U})$ such that $\mathsf{L} \leq \mathsf{V} \leq \mathsf{U}$ and $\mathsf{U}(s_0) - \mathsf{L}(s_0) \leq \varepsilon$
 1: $x_0 \leftarrow \textsc{InitVI}_\Phi$                          # Classic VI for $\Phi$
 2: $\mathsf{L}_0(\cdot) \leftarrow 0$, $\mathsf{U}_0(\cdot) \leftarrow \textsc{InitU}_\Phi$        # Initialize bounds
 3: $i \leftarrow 0$
 4: while $\mathsf{U}_i(s_0) - \mathsf{L}_i(s_0) > \varepsilon$ do
 5:     $i \leftarrow i + 1$
        # Convergent recommender procedure
 6:     $x_i \leftarrow \mathsf{VI}_\Phi(x_{i-1})$       # Classic converging VI for $\Phi$
 7:     $\sigma, \tau \leftarrow$ strategies inferred from $x_i$ (see Lemma 1)
        # Fixpoint updates for $\Phi$ on bounds in induced MDPs
 8:     $\hat{\mathsf{L}}_i \leftarrow \min(\mathsf{L}_i, \mathsf{Fix}_\Phi^{\mathsf{G}^{\sigma,\cdot}}(\mathsf{L}_{i-1}))$
 9:     $\hat{\mathsf{U}}_i \leftarrow \max(\mathsf{U}_i, \mathsf{Fix}_\Phi^{\mathsf{G}^{\cdot,\tau}}(\mathsf{U}_{i-1}))$
        # Additional local MDP reasoning, de- & inflate
10:     for each MSEC $E$ in $\mathsf{G}^{\cdot,\tau}$ do
11:         for all $s \in E$ do
12:             $\mathcal{U}_i(s) \leftarrow \max(\mathsf{stay}_{\mathsf{G}^{\cdot,\tau},\Phi}(E,s), \mathsf{exit}_{\hat{\mathsf{U}}_i}^{\Box}(E))$
13:             $\mathsf{U}_i(s) \leftarrow \min(\hat{\mathsf{U}}_i(s), \mathcal{U}_i(s))$
14:     for each MSEC $E$ in $\mathsf{G}^{\sigma,\cdot}$ do
15:         for all $s \in E$ do
16:             $\mathcal{L}_i(s) \leftarrow \min(\mathsf{stay}_{\mathsf{G}^{\sigma,\cdot},\Phi}(E,s), \mathsf{exit}_{\hat{\mathsf{L}}_i}^{\bigcirc}(E))$
17:             $\mathsf{L}_i(s) \leftarrow \max(\hat{\mathsf{L}}_i(s), \mathcal{L}_i(s))$
18:     Set $\mathsf{U}_i(s) \leftarrow \hat{\mathsf{U}}_i(s)$ and $\mathsf{L}_i(s) \leftarrow \hat{\mathsf{L}}_i(s)$ if not defined
```

To summarize, Lemma 4 identifies SEC as potential problems, while Lemma 5 provides us with a sound, conservative way of treating these "alarms". In the following, we combine this insight with classical value iteration, which gives us "completeness" in the form of termination guarantees. We emphasize that up until now, our reasoning holds for all fixpoint-linear objectives.

D. Algorithm

With these intuitions in mind, we now present our "local" solution approach in Algorithm 2. As in Algorithm 1, Lines 1, 6 and 7 describe the classic convergent VI procedure that is used to recommend strategies. However, the bounds L_i and U_i are not computed by solving an MDP. Instead, in Lines 8 and 9 we perform fixpoint updates on the bounds, i.e. Eq. (3) for total reward and Eq. (2) otherwise, to obtain $\hat{\mathsf{L}}_i$ and $\hat{\mathsf{U}}_i$. Since these might not converge, as shown in Example 1, we apply the additional local reasoning in the loops of Lines 12 and 16, obtaining $\mathcal{L}_i$ and $\mathcal{U}_i$, respectively.

We emphasize that this algorithm conceptually handles all the different objectives we consider. Independent of objective, the computation repeatedly (i) applies fixpoint updates and (ii) adjusts the approximations in problematic ECs so they are dependent on exit and staying values. Observe that we search for SECs in the MDP induced by the recommended strategies. This corresponds to the "guess" for SECs, as in [15]. Since the updates we perform are conservative, guessing wrongly does not hurt by the reasoning of the previous section, in particular Eqs. (4) and (5). It remains to discuss three sub-procedures.

Initializing Upper Bounds ($\textsc{InitU}_\Phi$*):* Firstly, we cannot initialize the upper bounds to ∞ everywhere, as this yields a fixpoint. Fortunately, for reachability, safety, and mean payoff, we can directly derive an upper bound (1 or the maximal occurring reward, respectively). For total reward, we can compute a finite upper bound on the total reward for all states as in [21]. See Appendix B for details.

Finding MSECs: We only need to find MSECs in the induced MDPs, which is straightforward in almost all cases: For reachability, safety, and mean payoff, those are exactly the MECs of the MDP by virtue of Lemma 2. For total reward, we need to employ further care: We excluded that states have a value of infinity *in the game*, however, Maximizer could obtain infinite reward in $\mathsf{G}^{\cdot,\tau}$ due to a sub-optimal Minimizer strategy. However, any MEC in the MDP either has value 0 or infinity, thus, still every MEC is MSEC in the MDP. For the Minimizer, observe that no state can have a value of infinity in $\mathsf{G}^{\sigma,\cdot}$ by our assumption: Against a sub-optimal strategy of Maximizer, Minimizer can achieve even smaller values. Thus, MSECs are all maximal end components that remain after removing all states with non-zero rewards, again by Lemma 2.

Computing the Staying Value: For all objectives except mean payoff, simple graph analysis is sufficient, see Appendix E. For mean payoff, an exact computation is possible, however we additionally discuss how we can obtain convergent under- and over-approximations from the iterates x_i. In particular, we can interleave the computation of staying values to finer and finer precisions with the propagation of bounds, similar to the *on-demand* adaptation of VI of [14]. When we replace the exact value with converging upper and lower bounds, the correctness and convergence guarantees do not change. We describe the details of both the exact as well as the approximate computation in Appendix E.

Now that the algorithm was intuitively explained, we formally state its effectiveness. The technical proof can be found in Appendix C-F.

Theorem 3. *Algorithm 2 is correct and terminates for any considered objective* Φ.

Note that Algorithm 2 again only relies on a (stable) recommender procedure and is not bound to value iteration.

VI. Discussion

To conclude, we provide several remarks on our algorithms for further insights and relate them to previous approaches.

A. Monotonicity

In both algorithms, we update bounds by additionally comparing with the previous lower respectively upper estimate to ensure monotonicity. This property is preferable, because a user of the algorithm naturally expects convergent anytime bounds to be monotonically improving. However, for the correctness and convergence this comparison is not necessary if the strategy recommender is stable.

B. Relation between Algorithms 1 and 2

Both algorithms rely on a recommender procedure that eventually suggests optimal strategies, and both of them use the suggested strategies to apply specialized reasoning to deal with spurious fixpoints in ECs. However, Algorithm 1 globally applies this reasoning by completely solving both induced MDPs every iteration. This allows us to transparently re-use existing methods for MDP solving. In contrast, Algorithm 2 performs the fixpoint updates on the bounds and then only applies the specialized reasoning locally to resolve cyclic dependencies. This way, Algorithm 2 avoids solving large parts of the MDP repeatedly which do not need specialized reasoning. However, it cannot immediately obtain the values when an optimal strategy is recommended, and might only converge in the limit.

C. Relationship to classical VI

For reachability, recall that the sequence $(x_i)_{i\in\mathbb{N}}$ already is a convergent lower bound on the value. Consequently, separate computation of lower bounds can be omitted, including considering the induced MDP. More generally, we can directly use x_i in place of L_i. The same holds for total reward [21]. Dually, for safety the upper bound computation can be omitted. Finally, for mean payoff, the values x_i/i of the classic VI can oscillate around the actual value, hence both bounds have to be computed in addition to x_i.

D. Relationship to [15]

In essence, [15] performs a structural analysis on MECs using the current approximations and then *deflates*, i.e. decreases the upper bounds to values of some Maximizer's (MEC-exiting) actions. From our perspective, it applies Minimizer's strategy τ and computes the best exits in the induced MDP. In the simpler case of reachability, the staying value is 0 in all non-trivial ECs and each MEC is also MSEC. Combining with the above insights, we obtain the algorithm of [15] as special case of Algorithm 2.

E. Relationship to [16]

The convergent anytime algorithm for reachability in SG from [15] has been extended to improve performance in [16]. In that work, the authors compute the over-approximation by constructing the MDP $\mathsf{G}^{\cdot,\tau}$ as in Algorithm 1. However, instead of solving the MDP completely, they over-approximate its value by turning it into a weighted graph and computing a widest path problem. The resulting values over-approximate the value in the MDP, and hence the values in the SG.

Eventually the weights in the graph are precise enough that the algorithm converges. Thus, to make Algorithm 1 practically more efficient, one can replace the complete MDP-solving also with the weighted-graph-based approximation from [16].

F. Relationship to EC collapsing

The presented insights, in particular Algorithm 2, capture and unify all previous treatment of problematic fixpoints in MDP. Indeed, Lemma 5 also holds for MDP. For the objectives reachability, safety, and mean payoff, Eqs. (6) and (7) actually yield an equality, as there is no second player that can interfere. Moreover, for these objectives on MDP, every EC is simple. In other words, Lemma 5 yields that the value of a (simple) EC exactly equals the maximum of staying and leaving value. Previous works [14] dealt with such objectives on MDP by collapsing ECs, i.e. replacing them with a single state that has all outgoing actions available and, in the case of mean payoff, a special "stay" action, which represents the value that can be obtained by remaining inside the EC. Observe the exact match with the above equations: By keeping all exiting actions, the best exit remains available to the player (at the time the EC is collapsed, the precise values of actions are not known, hence all exits need to be kept). Analogously, by adding the stay action, the staying value may still be obtained. Thus, applying the fixpoint update on the collapsed MDP takes the maximum over staying and all possible exits.

G. Discounted and Bounded Objectives

A prominent class of objectives are *discounting objectives*. For example, for discounted total reward, the value of a path is given by $\Phi(\rho) = \sum_{i=0}^{\infty}\gamma^i \mathsf{r}(\rho_i)$ for some discounting factor $\gamma < 1$. While our methods as presented are not immediately applicable – these objectives are not fixpoint-linear – there is a classical reduction for many discounted objectives to their non-discounted counterpart. For example, with total reward we (i) add a trap state $\underline{s}$ to the game with $\mathsf{r}(\underline{s}) = 0$, (ii) rescale every transition by $(1 - \gamma)$, and (iii) add a γ-probability transition to $\underline{s}$. By applying this technique, Algorithm 2 also is applicable to "discounting fixpoint-linear objectives", where $\mathsf{Fix}_\Phi(f, s) = \mathsf{opt}_{a \in \mathsf{A}(s)} \mathsf{off}_\Phi(s) + \gamma(s) \cdot \mathsf{Fix}_\Phi(f, s, a)$ with some (potentially state-dependent) discounting factor $\gamma(s) \in [0, 1]$. (Note that this requires a sensible reward value for $\underline{s}$.)

Similarly, *bounded objectives*, requiring, for example, that a goal state is reached before a time- or resource-bound is exceeded, can be handled by encoding the current usage into the state space. For step-/time-bounded objectives, no non-trivial ECs exist and classical VI already provides a complete solution: Iterating the Bellman operator K times yields the optimal value after K steps. However, for e.g. cost-bounded reachability there might be ECs comprising states with cost zero, potentially inducing a SEC that needs to be dealt with.

We highlight that both types of objectives are known to have a unique fixpoint and the problem of non-converging bounds does not arise. Yet, our framework provides a simple, objective independent proof for a wide class of objectives.

H. Other Variants of Total Reward

We now comment on adaptations to other variants of total reward. Aside from the "classical" total reward, where rewards are simply accumulated along the path, [21] introduces two other variants, which we refer to as TR_0 and TR_∞. Both are defined relative to a target set F, and $\mathsf{TR}_0(\rho) = \mathsf{TR}_\infty(\rho) = \sum_{i=1}^{k} \mathsf{r}(\rho_k)$ if $\rho \in \Diamond \mathsf{F}$ and k is the first time the target set is reached; otherwise $\mathsf{TR}_0(\rho) = 0$ and $\mathsf{TR}_\infty(\rho) = \infty$. Informally, we only obtain the accumulated reward if the target is reached, otherwise we get 0 or ∞, respectively. While TR_0 can be directly reduced to our variant [21, Sec. 4.3.3], $\mathsf{TR}_\infty(\rho)$ provides an interesting, "dual" variant. In particular, while for regular total reward it is Maximizer who wants to leave end components eventually, here Minimizer wants to leave, since otherwise ∞ is obtained.

As with the other objectives, we can again assume a canonical form. In particular, we require that no states have a value of infinity – exactly those from which Maximizer can ensure that the target states are not reached. We remark that nevertheless this objective is different from regular total reward, since Maximizer that can force Minimizer to take "large" exits, as also explained below.

We now discuss the simple adaptations required to show that our algorithms apply to TR_∞, too, in form of a "checklist".

Recommender: It is known that VI monotonically converges to the correct value from above [21]. To obtain Lemma 1, i.e. that value iteration recommends correct strategies, we can proceed completely analogous to the regular total reward proof (see Appendix C-A). In particular, by convergence guarantees, we automatically get that strategies are safe eventually.

This is already enough to show correctness and convergence for Algorithm 1! For Algorithm 2, only minor points remain – most of the reasoning in Section V-D is objective independent.

Fixpoint-Linear: This clearly holds for this objective.

Init and MSECs: Both initialization of value iteration (see Appendix B) and identification of MSECs is analogous to regular total reward.

As an interesting observation, since TR_∞ is fixpoint-linear, note that Lemma 5 holds. Since the staying value is ∞ (compared to 0 for regular total reward) and value iteration converges from above, we now only need to inflate instead of deflate, i.e. adjust Minimizers values inside SECs instead of Maximizer. Intuitively, Maximizer has the "momentum" in ECs and Minimizer is forced to eventually make a move. This is analogous to Remark 3, TR_∞ is like safety in this regard: a greatest fixpoint objective requiring only inflation. In contrast, reachability and "classical" total reward are least fixed point objectives and thus require only deflation.

VII. Conclusion

We have provided stopping criteria for SG with frequently used quantitative objectives. Our approach comes in two flavours. The first, generic algorithm allows to lift MDP-based reasoning to SG on a broad variety of objectives, requiring only very mild assumptions. The second, local-reasoning algorithm provides a unifying solution to previously as well as newly studied objectives, and, by encompassing previous solutions and tweaks, unifies a decade of advances on MDP and SG. More generally, we provide a solid theoretical foundation for obtaining converging bounds through value iteration for infinite-horizon objectives on stochastic games. We conjecture that it can be directly extended to ω-regular objectives (with the staying value of parity SG being computationally more complicated) or to multiple objectives (similar to multi-dimensional reachability [17]). Combining with other recent advances such as guessing [47] may also provide further insights (e.g. guessing the value of an entire SEC).

Since this work can serve as a basis for guaranteed and yet practically fast algorithms, it is desirable to extend the algorithms with adaptations of heuristics in order to foster the practical applicability. On the one hand, topological VI [48] or better initial values [22] are directly applicable. On the other hand, extending sound VI [12] or optimistic VI [13] to SG or using the widest path approach [16] seem promising future directions within reach. Besides, converging bounds allow for a sensible use of asynchronous value iteration and partial exploration, e.g. [20], [10], [15], which can then be guided by unreliable but potentially powerful learning heuristics. Once these improvements are properly put in place, their implementation and experimental comparison will be an interesting next step towards practically scalable solutions.

We conjecture that our reasoning can be directly extended to objectives requiring finite memory. Moreover, for many objectives requiring memory, e.g. ω-regular specifications, the required memory structure is known a-priori and the problem can be converted to a "memoryless" one by adding the memory to the state space of the game.

References

[1] L. S. Shapley, "Stochastic games," *Proc. Nat. Acad. Sci. U.S.A.*, vol. 39, pp. 1095–1100, 1953.

[2] F. Thuijsman and O. Vrieze, "The bad match, a total reward stochastic game," *Operations Research Spektrum*, vol. 9, pp. 93–99, 1987.

[3] D. Gillette, "Stochastic games with zero stop probabilities," *Contributions to the Theory of Games*, vol. 3, pp. 179–187, 1957.

[4] J. Mertens and A. Neyman, "Stochastic games," *Int. J. Game Theory*, vol. 10, pp. 53–66, 1981.

[5] A. Condon, "On algorithms for simple stochastic games," in *Advances In Computational Complexity Theory*, ser. DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 13. DIMACS/AMS, 1990, pp. 51–71.

[6] R. Bellman, "A Markovian decision process," *Journal of Mathematics and Mechanics*, vol. 6, no. 5, pp. 679–684, 1957.

[7] K. Chatterjee and T. A. Henzinger, "Value iteration," in *25 Years of Model Checking*, ser. Lecture Notes in Computer Science, vol. 5000. Springer, 2008, pp. 107–138.

[8] A. Hartmanns, S. Junges, T. Quatmann, and M. Weininger, "A Practitioner's Guide to MDP Model Checking Algorithms," in *TACAS*, ser. LNCS. Springer, 2023, To appear.

[9] S. Haddad and B. Monmege, "Interval iteration algorithm for MDPs and IMDPs," *Theor. Comput. Sci.*, vol. 735, pp. 111–131, 2018.

[10] T. Brázdil, K. Chatterjee, M. Chmelik, V. Forejt, J. Kretínský, M. Z. Kwiatkowska, D. Parker, and M. Ujma, "Verification of Markov decision processes using learning algorithms," in *ATVA*, ser. Lecture Notes in Computer Science, vol. 8837. Springer, 2014, pp. 98–114.

[11] C. Baier, J. Klein, L. Leuschner, D. Parker, and S. Wunderlich, "Ensuring the reliability of your model checker: Interval iteration for Markov decision processes," in *CAV (1)*, ser. Lecture Notes in Computer Science, vol. 10426. Springer, 2017, pp. 160–180.

[12] T. Quatmann and J. Katoen, "Sound value iteration," in *CAV (1)*, ser. Lecture Notes in Computer Science, vol. 10981. Springer, 2018, pp. 643–661.

[13] A. Hartmanns and B. L. Kaminski, "Optimistic value iteration," in *CAV (2)*, ser. Lecture Notes in Computer Science, vol. 12225. Springer, 2020, pp. 488–511.

[14] P. Ashok, K. Chatterjee, P. Daca, J. Kretínský, and T. Meggendorfer, "Value iteration for long-run average reward in Markov decision processes," in *CAV (1)*, ser. Lecture Notes in Computer Science, vol. 10426. Springer, 2017, pp. 201–221.

[15] J. Eisentraut, E. Kelmendi, J. Kretínský, and M. Weininger, "Value iteration for simple stochastic games: Stopping criterion and learning algorithm," *Inf. Comput.*, vol. 285, no. Part, p. 104886, 2022. [Online]. Available: https://doi.org/10.1016/j.ic.2022.104886

[16] K. Phalakarn, T. Takisaka, T. Haas, and I. Hasuo, "Widest paths and global propagation in bounded value iteration for stochastic games," in *CAV (2)*, ser. Lecture Notes in Computer Science, vol. 12225. Springer, 2020, pp. 349–371.

[17] P. Ashok, K. Chatterjee, J. Kretínský, M. Weininger, and T. Winkler, "Approximating values of generalized-reachability stochastic games," in *LICS*. ACM, 2020, pp. 102–115.

[18] M. Kwiatkowska, G. Norman, D. Parker, and G. Santos, "Prism-games 3.0: Stochastic game verification with concurrency, equilibria and time," in *CAV (2)*, ser. Lecture Notes in Computer Science, vol. 12225. Springer, 2020, pp. 475–487.

[19] H. Howard, *Dynamic Programming and Markov Processes*. MIT Press, 1960.

[20] H. B. McMahan, M. Likhachev, and G. J. Gordon, "Bounded real-time dynamic programming: RTDP with monotone upper bounds and performance guarantees," in *ICML*, ser. ACM International Conference Proceeding Series, vol. 119. ACM, 2005, pp. 569–576.

[21] T. Chen, V. Forejt, M. Z. Kwiatkowska, D. Parker, and A. Simaitis, "Automatic verification of competitive stochastic systems," *Formal Methods Syst. Des.*, vol. 43, no. 1, pp. 61–92, 2013.

[22] J. Kretínský, E. Ramneantu, A. Slivinskiy, and M. Weininger, "Comparison of algorithms for simple stochastic games," *Inf. Comput.*, vol. 289, no. Part, p. 104885, 2022. [Online]. Available: https://doi.org/10.1016/j.ic.2022.104885

[23] J. Kretínský and T. Meggendorfer, "Efficient strategy iteration for mean payoff in Markov decision processes," in *ATVA*, ser. Lecture Notes in Computer Science, vol. 10482. Springer, 2017, pp. 380–399.

[24] O. Friedmann, "An exponential lower bound for the parity game strategy improvement algorithm as we know it," in *Proceedings of the 24th Annual IEEE Symposium on Logic in Computer Science, LICS 2009, 11-14 August 2009, Los Angeles, CA, USA*. IEEE Computer Society, 2009, pp. 145–156. [Online]. Available: https://doi.org/10.1109/LICS.2009.27

[25] ——, "An exponential lower bound for the latest deterministic strategy iteration algorithms," *Log. Methods Comput. Sci.*, vol. 7, no. 3, 2011. [Online]. Available: https://doi.org/10.2168/LMCS-7(3:23)2011

[26] N. Balaji, S. Kiefer, P. Novotný, G. A. Pérez, and M. Shirmohammadi, "On the complexity of value iteration," in *ICALP*, ser. LIPIcs, vol. 132. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019, pp. 102:1–102:15.

[27] M. Z. Kwiatkowska, G. Norman, and D. Parker, "PRISM 4.0: Verification of probabilistic real-time systems," in *CAV*, ser. Lecture Notes in Computer Science, vol. 6806. Springer, 2011, pp. 585–591.

[28] C. Dehnert, S. Junges, J. Katoen, and M. Volk, "A storm is coming: A modern probabilistic model checker," in *CAV (2)*, ser. Lecture Notes in Computer Science, vol. 10427. Springer, 2017, pp. 592–600.

[29] T. Meggendorfer, "PET - A partial exploration tool for probabilistic verification," in *Automated Technology for Verification and Analysis - 20th International Symposium, ATVA 2022, Virtual Event, October 25-28, 2022, Proceedings*, ser. Lecture Notes in Computer Science, A. Bouajjani, L. Holík, and Z. Wu, Eds., vol. 13505. Springer, 2022, pp. 320–326. [Online]. Available: https://doi.org/10.1007/978-3-031-19992-9_20

[30] D. Andersson and P. B. Miltersen, "The complexity of solving stochastic games on graphs," in *ISAAC*, ser. Lecture Notes in Computer Science, vol. 5878. Springer, 2009, pp. 112–121.

[31] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, ser. Wiley Series in Probability and Statistics. Wiley, 1994. [Online]. Available: https://doi.org/10.1002/9780470316887

[32] C. Baier and J. Katoen, *Principles of model checking*. MIT Press, 2008.

[33] V. Forejt, M. Z. Kwiatkowska, G. Norman, and D. Parker, "Automated verification techniques for probabilistic systems," in *SFM*, ser. Lecture Notes in Computer Science, vol. 6659. Springer, 2011, pp. 53–113.

[34] J. Filar and K. Vrieze, *Competitive Markov decision processes*. Springer Science & Business Media, 2012.

[35] D. A. Martin, "The determinacy of blackwell games," *J. Symb. Log.*, vol. 63, no. 4, pp. 1565–1581, 1998.

[36] A. P. Maitra and W. D. Sudderth, "Finitely additive stochastic games with borel measurable payoffs," *Int. J. Game Theory*, vol. 27, no. 2, pp. 257–267, 1998. [Online]. Available: https://doi.org/10.1007/s001820050071

[37] T. Brázdil, A. Kucera, and P. Novotný, "Determinacy in stochastic games with unbounded payoff functions," in *MEMICS*, ser. Lecture Notes in Computer Science, vol. 7721. Springer, 2012, pp. 94–105.

[38] S. Kiefer, R. Mayr, M. Shirmohammadi, and D. Wojtczak, "On strong determinacy of countable stochastic games," in *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*. IEEE Computer Society, 2017, pp. 1–12. [Online]. Available: https://doi.org/10.1109/LICS.2017.8005134

[39] T. M. Liggett and S. A. Lippman, "Stochastic games with perfect information and time average payoff," *Siam Review*, vol. 11, no. 4, pp. 604–607, 1969.

[40] T. Brázdil, V. Brozek, A. Kucera, and J. Obdrzálek, "Qualitative reachability in stochastic BPA games," *Inf. Comput.*, vol. 209, no. 8, pp. 1160–1183, 2011. [Online]. Available: https://doi.org/10.1016/j.ic.2011.02.002

[41] U. Zwick and M. Paterson, "The complexity of mean payoff games on graphs," *Theor. Comput. Sci.*, vol. 158, no. 1&2, pp. 343–359, 1996.

[42] L. De Alfaro, "Formal verification of probabilistic systems," Ph.D. dissertation, Stanford University, 1997.

[43] C. Courcoubetis and M. Yannakakis, "The complexity of probabilistic verification," *J. ACM*, vol. 42, no. 4, pp. 857–907, 1995.

[44] A. J. Hoffman and R. M. Karp, "On nonterminating stochastic games," *Management Science*, vol. 12, no. 5, pp. 359–370, 1966.

[45] J. Kretínský and T. Meggendorfer, "Conditional value-at-risk for reachability and mean payoff in markov decision processes," in *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2018, Oxford, UK, July 09-12, 2018*, A. Dawar and E. Grädel, Eds. ACM, 2018, pp. 609–618. [Online]. Available: https://doi.org/10.1145/3209108.3209176

[46] T. Meggendorfer, "Risk-aware stochastic shortest path," in *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*. AAAI Press, 2022, pp. 9858–9867. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/21222

[47] K. Chatterjee, T. Meggendorfer, R. Saona, and J. Svoboda, "Faster algorithm for turn-based stochastic games with bounded treewidth," in *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, N. Bansal and V. Nagarajan, Eds. SIAM, 2023, pp. 4590–4605. [Online]. Available: https://doi.org/10.1137/1.9781611977554.ch173

[48] P. Dai, Mausam, D. S. Weld, and J. Goldsmith, "Topological value iteration algorithms," *J. Artif. Intell. Res.*, vol. 42, pp. 181–209, 2011.

[49] M. Akian, J. Cochet-Terrasson, S. Detournay, and S. Gaubert, "Solving multichain stochastic games with mean payoff by policy iteration," in *CDC*. IEEE, 2013, pp. 1834–1841. [Online]. Available: https://doi.org/10.1109/CDC.2013.6760149

Appendix A
Properties of Mean Payoff

We first give some intuition about the mean payoff objective. There are two fundamental notions, as described in e.g. [31], named *gain* and *bias*. Recall that each run of the system induces a sequence of rewards for mean payoff, say $10\ 10\ 5\ 10\ 5\ 3\ 5\ 3\ 5\ 3\ 5\ \cdots$. The gain of this run is 4 since it repeats 3 and 5 infinitely. However, in the beginning, before it "stabilizes" to this average of 4, the run achieves larger values, in particular several 10s. The bias of this run now equals the *total* deviation its gain, i.e. how its "transient" or "finite" behaviour differs from the infinite part (the gain). Note that almost all runs eventually end up in a MEC where then the gain is obtained. An alternative view to visualize the interplay of gain and bias is that, in the limit, the n-step total reward of a run equals n times the gain plus the bias, i.e. $v_n - n \cdot g + h = \mathcal{O}(1)$, where v_n denotes the n-step total reward of a run, g the gain and h the bias. The $\mathcal{O}(1)$ term (opposed to constant 0) stems from potential small fluctuations which we do not discuss in detail now. This view actually motivates the two approaches to compute mean payoff. Both considering the differences in increments as well as v_n/n yields the gain (where the former needs simple treatment of the $\mathcal{O}(1)$ term).

For Markov chains, we observe that BSCCs exactly mark the point where runs switch to the "infinite" regime, i.e. start "obtaining" their gain. On MDP, MECs generalize this idea: All states in a MEC have the same value since we can always move to any potential state from which an optimal value can be obtained with probability 1 and then replicate the strategy obtaining this value. Using this realization, [14] decompose the MDP into MECs, compute the optimal value achievable in each MEC (assuming the system is restricted to it), and then solve a "weighted reachability" query. Essentially, they treat each MEC as a soft target where the system can decide to *stay* and obtain the MECs value (replaying the optimal strategy) or move on, trying to get to a better MEC. Since under any strategy we always end up in MECs with probability 1, MECs are the only place where the system really "obtains" the mean payoff and anything prior essentially is a matter of reachability.

In the following, we lift some basic properties of mean payoff and several of these ideas to the world of SGs, with the additional hurdles posed by them. In particular, recall that we cannot simply identify regions where all states can obtain the same value (see Example 2), hence figuring out the optimal infinite play is significantly more involved. If we were given correct bounds (and regions where the infinite play takes place), we could apply the ideas of [14] to reduce the problem to reachability.

A. Rescaling the Reward

We restate (and prove) a standard result for mean payoff, mainly for illustrative reasons.

Lemma 6. *Let* $\mathsf{r} : \mathsf{S} \to \mathbb{R}$ *be a reward function for an SG* G*. For all states* $s \in \mathsf{S}$*, let* $\hat{\mathsf{r}}(s) = \mathsf{r}(s) \cdot a + b$ *be a rescaled reward function, where* $a, b \in \mathbb{R}$*. Then we have* $\mathsf{V}_{\mathsf{G}, mp_{\mathsf{r}}} \cdot a + b = \mathsf{V}_{\mathsf{G}, mp_{\hat{\mathsf{r}}}}$*.*

Proof: We first show below that for all paths $\rho \in \mathsf{Paths}_{\mathsf{G}}$, it holds that $mp_{\mathsf{r}}(\rho) \cdot a + b = mp_{\hat{\mathsf{r}}}(\rho)$.

Let $\rho \in \mathsf{Paths}_{\mathsf{G}}$ be an arbitrary path.

$$\begin{aligned}
mp_{\hat{\mathsf{r}}}(\rho) &= \liminf_{n\to\infty}\Big(\frac{1}{n}\sum_{j=0}^{n-1}\hat{\mathsf{r}}(\rho_j)\Big) && \text{(Def. of mean payoff)}\\
&= \liminf_{n\to\infty}\Big(\frac{1}{n}\sum_{j=0}^{n-1}\mathsf{r}(\rho_j)\cdot a + b\Big) && \text{(Def. of } \hat{\mathsf{r}}\text{)}\\
&= \left(\liminf_{n\to\infty}\Big(\frac{1}{n}\sum_{j=0}^{n-1}\mathsf{r}(\rho_j)\Big)\right)\cdot a + b \\
& && (a \text{ and } b \text{ independent of } j \text{ and } n)\\
&= mp_{\mathsf{r}}(\rho)\cdot a + b && \text{(By definition of mean payoff)}
\end{aligned}$$

The proof for the value, which is the expectation over the paths, follows directly from linearity of expectation.

$$\begin{aligned}
\mathsf{V}_{\mathsf{G}, mp_{\hat{\mathsf{r}}}} &= \sup\nolimits_\sigma \inf\nolimits_\tau \mathbb{E}^{\sigma,\tau}_{\mathsf{G},s}[mp_{\hat{\mathsf{r}}}] && \text{(Def. of value)}\\
&= \sup\nolimits_\sigma \inf\nolimits_\tau a \cdot \mathbb{E}^{\sigma,\tau}_{\mathsf{G},s}[mp_{\mathsf{r}}] + b && \text{(Linear expectation)}\\
&= \mathsf{V}_{\mathsf{G}, mp_{\mathsf{r}}} \cdot a + b && \text{(Def. of value)}
\end{aligned}$$

This concludes the proof. ■

This lemma allows us to rescale state rewards, as we can compute the mean payoff with the modified rewards and then obtain the original mean payoff by reversing the rescaling. This allows us to only consider positive rewards.

B. Bellman Equations for Mean Payoff

The mean payoff values are a fixpoint of Eq. (2). Observe that we do not use Eq. (2) to compute the mean payoff directly, but instead consider the total reward obtained from Eq. (3).

Lemma 7. *Let* G *be an SG and* Φ *a mean payoff objective. Then* $\mathsf{V}_{\mathsf{G},\Phi}$ *is a fixpoint of applying Eq.* (2)*.*

Proof: The proof essentially follows from the fact that mean payoff is prefix independent, i.e. the value obtained by a run does not change if we add or remove a finite prefix from it. Assume for contradiction that $\mathsf{V}_{\mathsf{G},\Phi}$ does not satisfy the equation, i.e. there exists a state s where the optimal expected value of $\mathsf{V}_{\mathsf{G},\Phi}$ over its successors does not equal its value. Moreover, let us assume for simplicity that s is a Maximizer state (the argument for Minimizer is analogous). The value of state s can be at most the value over its successors, since in s we can simply pick the optimal action once and then replicate the witness strategies in each of its successors. Dually, the value of s can not be larger, since the witness strategy for this value has to achieve, on average, the value in s. Both of the statements can formally be obtained from the existing theory on mean payoff applied to the Markov chain induced by the witness strategies, see e.g. [31, Thm. 8.2.6], which shows that $\mathsf{V}_{\mathsf{M},\Phi}(s) = \sum_{s' \in \mathsf{S}} \delta(s, s') \cdot \mathsf{V}_{\mathsf{M},\Phi}(s')$ by Eq. (8.2.11). (We remark that [31] refers to mean payoff as *gain* and to Markov chains with rewards as *Markov reward process* (MRP).) ■

C. Convergence of Value Iteration

Here we show that iterating a total reward value iteration according to Eq. (3) allows us to infer the mean payoff in the limit. While [7, Sec. 5.2] proposed to extend the reasoning from [41, Sec. 2] to account for probabilities, we provide a much simpler proof by reducing to MDP and using classical results from [31].

Lemma 8. *Let* G *be an SG and* mp_{r} *a mean payoff objective. Let* $x_0 : \mathsf{S} \to \mathbb{R}_{\geq 0}$ *assign 0 to every state, and* $(x_i)_{i \in \mathbb{N}_0}$ *be the sequence of assignments computed by iterating Eq.* (3)*, using* r *as the reward function. Then we have*

$$\lim_{k\to\infty} x_k/k = \mathsf{V}_{\mathsf{G},mp_{\mathsf{r}}}$$

Proof: Recall that there exists a sequence of strategies σ_k that achieve a k-step total reward of at least x_k against any minimizer strategy. These strategies can be obtained by e.g. simply following the witnesses of Eq. (3). For a formal proof, we extend the induction in [21, Appendix B.1] to also include k-step-bounded optimality of these witnesses, similar to [32, Remark 10.104]. Moreover, let τ^* be a (memoryless, deterministic) mean payoff optimal minimizer strategy in G and fix $\mathcal{M} = \mathsf{G}^{\cdot,\tau^*}$ the induced MDP.

Assume for contradiction that $\limsup_{k\to\infty} x_k/k > \mathsf{V}_{\mathsf{G},mp_{\mathsf{r}}}$. (The dual case for $\liminf$ follows analogously.) Clearly, σ_k achieves a total reward of at least x_k in $\mathcal{M}$, too (since τ^* may be sub-optimal for total reward). In other words, the optimal k-step total reward in the game x_k is a lower bound on the optimal k-step total reward in $\mathcal{M}$, denoted by v_k. By applying [31, Thm. 9.4.1 b)][(4)] to $\mathcal{M}$, we get that $\lim_{k\to\infty} v_k/k = \mathsf{V}_{\mathcal{M},mp_{\mathsf{r}}}$. Together, $\lim_{k\to\infty} v_k/k = \mathsf{V}_{\mathcal{M},mp_{\mathsf{r}}} \geq \limsup_{k\to\infty} x_k/k > \mathsf{V}_{\mathsf{G},mp_{\mathsf{r}}}$, i.e. there exists a strategy in $\mathcal{M}$ that achieves a strictly higher mean payoff against τ^* than $\mathsf{V}_{\mathsf{G},mp_{\mathsf{r}}}$, contradicting optimality of τ^*. ∎

Appendix B
Initializing Value Iteration

For lower bounds, we observe that 0 is correct for all considered objectives: For reachability and safety this follows immediately, for total reward since we assumed rewards to be non-negative, and for mean payoff we have argued that w.l.o.g. we can as well assume that all rewards are non-negative (Lemma 6). To obtain upper bounds, reachability and safety are clearly bounded by 1, while mean payoff is bounded by the maximal occurring reward $\max_{s\in\mathsf{S}} \mathsf{r}(s)$. Total reward is a bit more involved. We aim to compute a rational number that is greater than the highest possible non-infinite total reward. To this end, we extend the procedure from [11, Sec. 3.3] to SG. In essence, the idea is as follows: Let $p_{\min}$ denote the minimum transition probability in the SG and let s be any state with non-infinite value. Fix optimal strategies (σ^*, τ^*). Since s has a finite value, the probability to eventually reach a region where zero rewards are obtained is equal to 1 in $\mathsf{G}^{\sigma^*,\tau^*}$. From classical observations on Markov chains [32],

(4)Note that v_n there denotes the optimal n-step total reward and g^* the optimal mean payoff in $\mathcal{M}$.

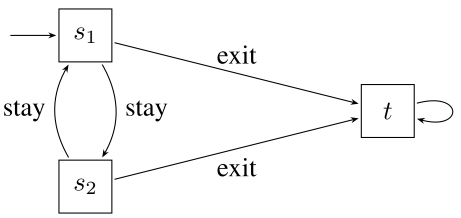


Figure 5: An example of an SG (also MDP). Considering the reachability query with target $\mathsf{F} = \{t\}$, following "safe" actions is not necessarily an optimal strategy.

we thus know that there is a probability of at least $p_{\min}^{|\mathsf{S}|}$ for this to happen within $|\mathsf{S}|$ steps (the smallest possible probability of a path of length n). This path can obtain at most $R_{\max} = |\mathsf{S}| \cdot \max_{s\in\mathsf{S}} \mathsf{r}(s)$ reward. Hence, the maximum total reward can be bounded by $\sum_{i=1}^{\infty} (1 - p_{\min}^{|\mathsf{S}|})^i \cdot R_{\max} < \infty$. In [11, Sec. 3.3], more refined ways are presented.

Appendix C
Proofs

A. Proof of Lemma 1

In this section, we make use of the notion of *fixpoint-linear* objectives, which we introduce in Definition 2. Effectively, we require that $\mathsf{V}_{\mathsf{G},\Phi}(s) = \mathsf{off}_\Phi(s) + \mathsf{opt}_{a\in\mathsf{A}(s)} \mathsf{V}_{\mathsf{G},\Phi}(s,a)$, i.e. the value function satisfies an (affine) linear equation.

We repeat the notion of *safe* strategies, which are those that take actions optimal w.r.t. the value.

Definition 5 ([30])**.** A strategy profile $\pi = (\sigma, \tau)$ is *safe* (for Φ) if $\pi(s) \in \arg\mathrm{opt}_{a\in\mathsf{A}(s)} \mathsf{V}_{\mathsf{G},\Phi}(s,a)$ for all states $s \in \mathsf{S}$.

Following safe strategies for any finite time is correct:

Lemma 9. *Suppose* Φ *is fixpoint-linear,* π *a safe strategy for* Φ*, and* π^* *an optimal strategy. Following* π *for* k *steps and then* π^* *yields optimal values.*

Proof: We proceed by induction over k, showing the statement for all states of G. Clearly, the statement holds for $k = 0$ – we simply follow an optimal strategy. From k to $k+1$, fix a state s. By assumption, $\mathsf{V}_{\mathsf{G},\Phi}(s) = \mathsf{off}_\Phi(s) + \mathsf{opt}_{a\in\mathsf{A}(s)} \mathsf{V}_{\mathsf{G},\Phi}(s,a)$. Since π is safe, $\pi(s)$ is such a witness, i.e. $\mathsf{V}_{\mathsf{G},\Phi}(s) = \mathsf{off}_\Phi(s) + \mathsf{V}_{\mathsf{G},\Phi}(s,\pi(s))$. By definition, $\mathsf{V}_{\mathsf{G},\Phi}(s,\pi(s)) = \sum_{s'\in\mathsf{S}} \delta(s,\pi(s),s') \cdot \mathsf{V}_{\mathsf{G},\Phi}(s')$. Inserting the induction hypothesis for $\mathsf{V}_{\mathsf{G},\Phi}(s')$ yields that following π for k steps and then π^* is optimal. Thus, following π for $k+1$ steps in s is optimal, too, concluding the proof. ∎

Unfortunately, following a safe strategy *forever* may not be correct. In other words, "safety" is only necessary but not sufficient for optimality.

Example 5. Consider the system in Fig. 5 (which is an MDP). We consider reachability with target $\mathsf{F} = \{t\}$. Clearly, all states have a value of 1. However, playing the action "stay" in both s_1 and s_2 reduces their reachability probability to 0. Thus, simply picking any optimal action $a \in \arg\mathrm{opt}_{a\in\mathsf{A}(s)} \mathsf{V}_{\mathsf{G},\Phi}(s,a)$ is not guaranteed to yield a correct strategy. Observe that even the value iterates have $x_i(s) = 1$ for all states s and $i \geq 2$. Consequently, we cannot choose an optimal action solely based on the current iterates, either.

We need to ensure that strategies additionally make "progress" instead of being stuck in spurious fixpoints. This is, for example, ensured by our proposed construction of only switching action if there is a strictly better alternative, as we show in the following. Alternatively, one can choose among optimal actions uniformly at random, which also ensures that a "progressing" action is chosen with non-zero probability. For a slightly different perspective, we direct the interested reader to [30], which ensures progress by choosing actions minimizing the distance to the goal, using an "attractor" computation. (In the context of reachability, [30] identifies "stopping" as additional criterion to be satisfied on top of safety, capturing this idea of progressing towards the target.)

Before proceeding with the main proof, we show a useful auxiliary statement for a broad variety of objectives. Intuitively, we show that strategies produced by Lemma 1 make progress towards "the right" BSCCs.

Lemma 10. *Let* VI *a Bellman update of the following form: We have* $\mathsf{VI}(x)(s) = c(s)$ *for* special states $s \in C$ *which are all absorbing, with* $c(s) \geq 0$ *a constant. For other states, we have* $\mathsf{VI}_\Phi(x)(s) = \mathsf{r}(s) + \mathsf{opt}_{a \in \mathsf{A}(s)} x(s,a)$ *with* $\mathsf{r}(s) \geq 0$. *Moreover, suppose that the iterates* x_i, *i.e.* $x_{i+1} = \mathsf{VI}_\Phi(x_i)$, *are either (i) monotonically increasing or (ii) monotonically decreasing.*

Let $\pi_i = (\sigma_i, \tau_i)$ *be the strategies obtained in step* i *as defined by Lemma 1. Then, every BSCC* R *in* G^{π_i} *either exists in all* G^{π_j} *with* $j \leq i$ *or* $\mathsf{r}(s) > 0$ *for some* $s \in R$.

Proof: Suppose a new BSCC R emerges in step k. Since special states are absorbing and never change their value, all states in this BSCC are not in C. Thus, there exists a state $s \in R$ where one player changed strategy due to a *strict* improvement. In other words, $x_{k-1}(s) \prec_s \mathsf{r}(s) + x_{k-1}(s, \pi_k(s)) = x_k(s)$. (The important fact is that $x_{k-1}(s) \neq x_k(s)$.) Moreover, $(s, \pi_{k-1}(s))$ exits the EC corresponding to R, i.e. $(R, \{\pi_k(s') \mid s' \in R\})$ (otherwise R would already exist in $\mathsf{G}^{\pi_{k-1}}$).

We first treat the case of monotonically increasing x_i, i.e. $x_i \leq x_{i+1}$ (which is the case for reachability, total reward, and mean payoff). Then, we have

$$\begin{aligned} x_k(s) &= \mathsf{VI}(x_{k-1})(s) \\ &= \mathsf{r}(s) + \sum\nolimits_{s' \in R} \delta(s, \pi_k(s), s') \cdot x_{k-1}(s') \\ &= \begin{matrix} \mathsf{r}(s) + \delta(s, \pi_k(s), s) \cdot x_{k-1}(s) + \\ \sum\nolimits_{s' \in R, s' \neq s} \delta(s, \pi_k(s), s') \cdot x_{k-1}(s') \end{matrix} \\ &\leq \begin{matrix} \mathsf{r}(s) + \delta(s, \pi_k(s), s) \cdot x_{k-1}(s) \\ + \sum\nolimits_{s' \in R, s' \neq s} \delta(s, \pi_k(s), s') \cdot x_k(s'). \end{matrix} \end{aligned}$$

We can repeatedly insert the definition of $x_k(s') = \mathsf{r}(s') + x_{k-1}(s', \pi_k(s'))$, split by the probability of transitioning from s' to s and to other states s'', and again use the monotonicity of x_i, i.e. $x_{k-1}(s'') \leq x_k(s'')$. In other words, we consider all potential finite paths from s back to itself inside this EC, which in sum have a probability of 1. Formally, let $P = \{\rho \mid s_1 = s \wedge s_n = s \wedge s \neq s_i \text{ for all } 1 < i < n\}$ and set $\mathsf{r}(\rho) = \sum \mathsf{r}(\rho_i)$ the sum of rewards along a path. Then, by above reasoning,

$$\begin{aligned} x_k(s) &\leq \sum\nolimits_{\rho \in P} \mathrm{Pr}^{\pi_i}_{\mathsf{G},s}[\rho] \cdot (\mathsf{r}(\rho) + x_{k-1}(s)) \\ &= \left(\sum\nolimits_{\rho \in P} \mathrm{Pr}^{\pi_i}_{\mathsf{G},s}[\rho] \cdot \mathsf{r}(\rho)\right) + x_{k-1}(s). \end{aligned}$$

We end up with $x_k(s) \leq W + x_{k-1}(s)$, where W is the weighted sum of all rewards obtained along these paths.

Observe that $W = 0$ if and only if $\mathsf{r}(s) = 0$ for all $s \in R$. Thus, when $W = 0$, then $x_k(s) \leq x_{k-1}(s)$ and, by monotonicity, $x_{k-1}(s) \leq x_k(s)$, a contradiction to $x_{k-1}(s) \neq x_k(s)$. In summary, we can only have a BSCC if $\mathsf{r}(s) > 0$ or this BSCC already exists from step 0 (i.e. we never switched action).

For the case of monotonically decreasing x_i (i.e. safety), the above reasoning still applies. By considering the paths inside R, we dually get $x_k(s) \geq W + x_{k-1}(s)$. By monotonicity, $x_k(s) \leq x_{k-1}(s)$. Thus, if $W = 0$, $x_k(s) = x_{k-1}(s)$, contradicting strict improvement. ■

We remark that a similar statement can be derived when choosing uniformly at random among all optimal actions. Then, we only get BSCCs if *all* exiting actions are strictly worse. However, intuitively, this cannot happen if $\mathsf{r}(s) = 0$ inside the BSCC, since a strictly better value cannot appear "out of thin air".

With these tools at hand, we are ready to prove Lemma 1, which we restate here for readability.

Lemma 1. *Fix an initial pair of strategies* (σ_0, τ_0). *For any Maximizer state* s, *define* $A_i(s)$ *as the set of actions in state* s *that witness the update of the iterated vector* x_i, *i.e. all actions in* $\arg\mathrm{opt}_{a \in \mathsf{A}(s)} x_i(s,a)$. *If* $\sigma_{i-1}(s) \in A_i(s)$, *define* $\sigma_i(s) = \sigma_{i-1}(s)$. *Otherwise, pick any* $a \in A_i(s)$ *and set* $\sigma_i(s) = a$. *We analogously define* τ_i. *Then, the resulting sequence* (σ_i, τ_i) *is a stable strategy recommender for all considered objectives.*

Proof: For readability, we write V instead of $\mathsf{V}_{\mathsf{G},\Phi}$.

Strategies are safe: First, we show that eventually the actions picked by either strategy are safe, i.e. there exists k_0 such that $\pi_k(s) \in \arg\mathrm{opt}_{a \in \mathsf{A}(s)} \mathsf{V}(s,a)$ for all $k \geq k_0$. (Note that this does *not* imply that the strategies remain constant.) This follows immediately from the convergence guarantees for reachability and safety [7, Section 3], as well as total reward [21, Appendix B.1]. There, we know that $\lim_{i \to \infty} x_i(s) = \mathsf{V}(s)$. Thus, suppose that $\pi_i(s)$ would pick a suboptimal action b infinitely often. This means that $x_i(s, a^*) \prec x_i(s,b)$ for all optimal actions $a^* \in \arg\mathrm{opt}_{a \in \mathsf{A}(s)} \mathsf{V}(s,a)$. However, since $x_i(s,b) \to \mathsf{V}_{\mathsf{G},\Phi}(s,b)$ by the convergence guarantees, we obtain a contradiction.

In the case of mean payoff, we need to argue separately, since the iterates do not directly yield the value. We employ reasoning similar to the one in the proof of Lemma 8. Suppose that a suboptimal action b is selected infinitely often. This means that by using action b, we can obtain at least as much k-step total reward as by using any optimal action a^* for arbitrarily large k. (This follows from results on total reward iteration.) Thus, $\limsup_{k\to\infty} \mathsf{V}_{\mathsf{G},\mathsf{TR}}(s,b)/k \geq \mathsf{V}(s,a^*) = \mathsf{V}(s)$. Now, observe the left hand side converges to the optimal mean payoff achieved under b (see Lemma 8 for further details), contradicting the sub-optimality of b.

Strategies do not follow spurious fixpoints: Now, we know that strategies are safe. However, this does not exclude the issues outlined in Example 5. Using Lemma 10, we now show that in all BSSCs induced by these strategies they eventually do actually obtain the optimal value. Formally, let R a BSCC in G^{π_i}. We show that $\mathsf{V}_{\mathsf{G}^{\pi_i},\Phi}(s) = \mathsf{V}_{\mathsf{G},\Phi}(s)$ for all $s \in R$ and large enough i. We consider each objective separately.

We begin with reachability. We prove that π_i eventually only induces BSCCs where either all states have value zero or equal a target state. The claim then follows immediately. By Lemma 10, the only non-target BSCCs that can exist are those which exist from the beginning. Thus, let R a BSCC in G^{π_0} and $R' = \{s \mid s \in R \wedge \mathsf{V}(s) > 0\} \subseteq R$ the states with non-zero value. We argue that a state s in R' eventually switches action. We know that the iterates x_i converge to the true value (which is > 0 in all states of R'). Consequently, there has to be a step k where for the first time any state of R' changes from $x_{k-1}(s) = 0$ to $x_k(s) > 0$ (this could happen for multiple states at once). However, this necessarily implies that state s changes action: By assumption, at step $k-1$, all states $s' \in R$ have a value of $x_{k-1}(s') = 0$, consequently $\sum_{s'\in R} \delta(s, \pi_{k-1}(s), s') \cdot x_{k-1}(s') = 0$. Thus, the witness action necessarily needs to exit R with positive probability. In summary, for large enough i, there are no non-target BSCCs under π_i where $\mathsf{V}(s) > 0$. For safety, the same reasoning can be applied. (For these two objectives, is is known that this already happens within at most $|\mathsf{S}|$ steps.)

For total reward, we show that eventually all BSCCs R in G^{π_i} have $\mathsf{V}(s) = 0$ for all states $s \in R$. To this end, suppose that the π_i we consider already are safe. Let R be a BSCC of G^{π_i}. Suppose that $\mathsf{r}(s) > 0$ for some state $s \in R$. By Lemma 9, both players are content with remaining in this BSCC arbitrarily long. This includes visiting s arbitrarily often, implying that the value in this BSCC is infinite (otherwise, Minimizer would eventually leave). However, this is excluded by assumption. Now, we follow the same reasoning as above: If there initially is a BSCC with states of non-zero value, this BSCC eventually dissolves. Together, the only BSCCs that remain are those with both state reward zero and a value of zero, proving the claim.

Finally, for mean payoff, this immediately follows from the reasoning of Lemma 8: All actions which are k-step total reward optimal for large enough k are mean payoff optimal.

Strategies behave correctly on transient states: We have shown that for large enough i the strategies π_i "do the correct thing" once they reach BSCCs. It remains to argue that we do not miss out on anything on the way. To prove this, first observe that eventually the strategies π_i we consider are safe by above reasoning. Moreover, on all induced BSCCs the strategy π_i is optimal. Thus, there exists an optimal strategy π^* which behaves like π_i on all BSCCs induced by π_i. Formally, let $\mathcal{B}_i$ the set of all BSCCs in G^{π_i}. Then, we have $\pi_i(s) = \pi^*(s)$ for all $s \in R, R \in \mathcal{B}_i$. Next, observe that the probability to reach BSCCs in finitely many steps approaches 1, i.e. $\lim_{k\to\infty} \mathrm{Pr}_{\mathsf{G}^{\pi_i},s}[\Diamond^{\leq k}\mathcal{B}_i] = 1$. Thus, we apply Lemma 9 for arbitrarily large k to π_i and π^*. By the above reasoning, the probability not to be in a BSCC after k steps (i.e. switching from π_i to π^*) tends to zero, thus the value achieved by the strategy composed of π_i and π^* after k steps tends to the value achieved by π_i. Formally, set π_i^k the strategy following π_i for k steps and then π^* as defined above. Then,

$$\mathsf{V}_{\mathsf{G}^{\pi_i^k},\Phi}(s) = R_i^k + \sum\nolimits_{s'\in\mathsf{S}} \mathrm{Pr}^{\pi_i^k}_{\mathsf{G},s}[\rho_k = s'] \cdot \mathsf{V}_{\mathsf{G}^{\pi^*},\Phi}(s'),$$

where R_i^k are the weighted rewards obtained by π_i in the first k steps (which is optimal by Lemma 9) and $\mathrm{Pr}^{\pi_i^k}_{\mathsf{G},s}[\rho_k = s']$ denotes the probability of ending up in s' after k steps. (This can also be directly obtained by simply considering all finite paths of length k.) To conclude, observe that by previous argument for large enough k the probability of ending up in a BSCC approaches 1, where $\mathsf{V}_{\mathsf{G}^{\pi_i},\Phi} = \mathsf{V}_{\mathsf{G}^{\pi^*},\Phi} = \mathsf{V}_{\mathsf{G},\Phi}$. ∎

B. Proof of Lemma 2

Lemma 2. *Let f an assignment and $E = (R, B)$ an EC. Then*

$$f(s) = \mathsf{off}_\Phi(s) + f(s,a) \textit{ for all } s \in R, a \in \mathsf{A}(s) \cap B,$$

implies $\mathsf{off}_\Phi(s) = 0$ for all $s \in R$.

Proof: Fix E and f as in the assumptions. Choose an arbitrary pair of states s and t. We show that $f(s) = f(t)$. Since E is an EC, there exists a pair of memoryless deterministic strategies that reach t almost surely, using only actions of E. Fix such a strategy pair $\pi = (\sigma, \tau)$ (these strategies do not need to be optimal for Φ). Then $f(s) = \mathsf{off}_\Phi(s) + \delta(s, \pi(s), t) \cdot f(t) + \sum_{s'\in R\setminus\{t\}} \delta(s, \pi(s), s') \cdot s'$ with $\mathsf{off}_\Phi(s) \geq 0$. Since the state t is reached almost surely, by repeatedly inserting this equation for $f(s')$ we end up with $f(s) = V + f(t)$, where $V \geq 0$. More formally, for a path $\rho = s_1 a_1 \cdots a_{n-1} s_n$, set $\mathsf{off}_\Phi(\rho) = \sum_{i=1}^n \mathsf{off}_\Phi(s_i)$ the sum of all rewards obtained along ρ and $\delta(\rho) = \prod_{i=1}^{n-1} \delta(s_i, a_i, s_{i+1})$ its probability. Define $P = \{\rho \mid s_1 = s \wedge s_n = t \wedge t \neq s_i \text{ for all } i < n\}$ the set of all paths from s to t. Then, $f(s) = \sum_{\rho\in P} \delta(\rho) \cdot (\mathsf{off}_\Phi(\rho) + f(t))$. We have that $\sum_{\rho\in P} \delta(\rho) = 1$ and $\mathsf{off}_\Phi(\rho) \geq 0$ for all ρ. Together, we get the desired result, namely that $f(s) = f(t) + \sum_{\rho\in P} \delta(\rho) \cdot \mathsf{off}_\Phi(\rho) \geq f(t)$. We can apply the same idea to "get back" to s and obtain $f(s) = V + V' + f(s)$, showing that $V = V' = 0$ and thus $\mathsf{off}_\Phi(s) = 0$ (note that we exclude $f(s) = \infty$). As an immediate consequence, we also get that $f(s) = f(t)$. ∎

C. Proof of Corollary 1

Corollary 1. *Let f an assignment and E an EC. Then, E is a SEC for f if and only if (i) $f(s) = f(s')$ and (ii) $\mathsf{off}_\Phi(s) = 0$ for all $s, s' \in E$.*

Proof: Note that the backward direction follows immediately. The forward direction follows exactly as in Lemma 2. Since E is SEC for f, we can again follow a "loop" from state s to t to derive that $\mathsf{off}_\Phi(s) = 0$, which implies $f(s) = f(t)$. ■

D. Proof of Lemma 3

Lemma 3. *Let f a fixpoint for Fix_Φ. If $f \not\equiv \mathsf{V}_{\mathsf{G},\Phi}$, then there exists a SEC E for f where $f(s) \neq \mathsf{V}_{\mathsf{G},\Phi}(s)$ for all $s \in E$.*

We prove a stronger statement, implying the above.

Lemma 11. *Let f a fixpoint for Fix_Φ. Set $d(s) = f(s) - \mathsf{V}_{\mathsf{G},\Phi}(s)$ its difference to the value and $d^+ = \max_{s \in S} d(s)$ and $d^- = \min_{s \in S} d(s)$ the maximum and minimum difference. There exist Φ-SECs E^+ and E^- with $d(s) = d^+$ and $d(s) = d^-$ for all states in E^+ and E^- respectively.*

Proof: For readability, we write V instead of $\mathsf{V}_{\mathsf{G},\Phi}$. Let f as in the assumptions. Observe that $d(s,a) = \sum_{s' \in S} \delta(s,a,s')d(s) = \sum_{s' \in S} \delta(s,a,s')(f(s') - \mathsf{V}(s')) = f(s,a) - \mathsf{V}(s,a)$. Let $d^+ = \max_{s \in S} d(s)$ the maximal difference between f and V. Define $S^+ = \{s \mid d(s) = d^+\}$ all states witnessing the maximal difference. (Note that d^+ might be zero, e.g. if $f \equiv \mathsf{V}$, or even smaller than zero if $f < \mathsf{V}$.)

We show that S^+ necessarily contains at least one SEC for f. In particular, we prove that for every $s \in S^+$, there exists an action $a \in \mathsf{A}(s)$ such that (i) (s,a) does not exit S^+ and (ii) a is a witness for the SEC condition. Thus, fix an arbitrary state $s \in S^+$. Observe that $d(s,a) = \sum_{s' \in S} \delta(s,a,s')(f(s') - \mathsf{V}(s'))$. By definition of d^+, we have $f(s) - \mathsf{V}(s) \leq d^+$ for all states s. Thus, the above sum is strictly smaller than d^+ exactly if at least one successor s' of (s,a) has $d(s') < d^+$, meaning $s' \notin S^+$. In other words, (s,a) exits S^+ iff $d(s,a) < d^+$. We proceed by a case distinction on the owner of s.

Maximizer: By definition, V is a fixpoint for Fix_Φ, i.e. $\mathsf{V}(s) = \mathsf{off}_\Phi(s) + \max_{a \in \mathsf{A}(s)} \mathsf{V}(s,a)$. For every action $a' \in \mathsf{A}(s)$, we thus have $\mathsf{V}(s) \geq \mathsf{off}_\Phi(s) + \mathsf{V}(s,a')$: The action a' can at most be the optimal one in the maximum. Together, for every action $a' \in \mathsf{A}(s)$ we have $d(s) = f(s) - \mathsf{V}(s) \leq f(s) - \mathsf{V}(s,a') - \mathsf{off}_\Phi(s)$. Recall that f is Fix_Φ fixpoint; let a_s the witness action for $f(s) = \mathsf{off}_\Phi(s) + \max_{a \in \mathsf{A}(s)} f(s,a)$, i.e. $f(s) = \mathsf{off}_\Phi(s) + f(s,a_s) = \mathsf{Fix}_\Phi(f,s)$. Suppose that (s,a_s) exits S^+. By the above argument, we have $d(s,a_s) < d^+$. Since $s \in S^+$, we have $d(s) = d^+$. Together, $d(s,a_s) < d(s) \leq f(s) - \mathsf{V}(s,a_s) - \mathsf{off}_\Phi(s) = f(s,a_s) - \mathsf{V}(s,a_s) = d(s,a_s)$, a contradiction. Thus, (s,a_s) does not exit S^+. Moreover, by choice of a_s we have $f(s) = f(s,a_s) = \mathsf{Fix}_\Phi(f,s)$.

Minimizer: Since $f(s) = \mathsf{off}_\Phi(s) + \min_{a \in \mathsf{A}(s)} f(s,a)$, we analogously get $d(s) = f(s) - \mathsf{V}(s) \leq \mathsf{off}_\Phi(s) + f(s,a') - \mathsf{V}(s)$ for any $a' \in \mathsf{A}(s)$. As above, let a_s such that $\mathsf{V}(s) = \mathsf{off}_\Phi(s) + \mathsf{V}(s,a_s)$ and assume (s,a_s) exits S^+. Then $d(s,a_s) < d(s) \leq f(s,a_s) - \mathsf{V}(s,a_s) = d(s,a_s)$, again a contradiction.

Quite surprisingly, even though a_s was chosen as arbitrary witness for $\mathsf{V}(s)$, it turns out that $f(s,a_s) = \mathsf{Fix}_\Phi(f,s)$, too: First, observe that $f(s) = \mathsf{V}(s) + d^+$ for all states in S^+. Since (s,a_s) does not exit S^+, this in particular holds for all successors of (s,a_s). Moreover, clearly $f(s) = \mathsf{off}_\Phi(s) + \min_{a \in \mathsf{A}(s)} f(s,a) \leq \mathsf{off}_\Phi(s) + f(s,a_s)$. Together, we get

$$\begin{aligned} f(s) &= \mathsf{V}(s) + d^+ = \mathsf{off}_\Phi(s) + \mathsf{V}(s,a_s) + d^+ \\ &= \mathsf{off}_\Phi(s) + \left(\sum\nolimits_{s' \in S} \delta(s,a_s,s')\mathsf{V}(s')\right) + d^+ \\ &= \mathsf{off}_\Phi(s) + \sum\nolimits_{s' \in S} \delta(s,a_s,s')(\mathsf{V}(s') + d^+) \\ &= \mathsf{off}_\Phi(s) + \sum\nolimits_{s' \in S} \delta(s,a_s,s') f(s') \\ &= \mathsf{off}_\Phi(s) + f(s,a_s). \end{aligned}$$

In short, $\mathsf{Fix}_\Phi(f,s) = f(s) = f(s,a_s) + \mathsf{off}_\Phi(s)$.

Together, every state has an action a_s that (i) does not exit S^+ and (ii) witnesses $f(s) = \mathsf{off}_\Phi(s) + \mathsf{opt}^s_{a \in \mathsf{A}(s)} f(s,a) = \mathsf{Fix}_\Phi(f,s)$. To conclude, consider the game where we restrict to S^+ and set $\mathsf{A}(s) = \{a_s\}$, which is well defined by (i). This game contains at least one EC, with $\mathsf{off}_\Phi(s) = 0$ by Lemma 2, and, by (ii), this is an SEC in G. Moreover, since this SEC is a subset of S^+, we have that all states in SEC have $d(s) = d^+$, i.e. this EC qualifies for E^+.

The proof for d^- and E^- follows analogously by "exchanging" the proofs for Maximizer and Minimizer. ■

E. Proof of Lemma 5

Lemma 5. *Fix a memoryless Minimizer strategy τ. Let $E = (R,B)$ be a SEC and f an upper bound on the value $\mathsf{V}_{\mathsf{G},\Phi}$. Then, we have for every $s \in R$*

$$\mathsf{V}_{\mathsf{G},\Phi}(s) \leq \max(\mathsf{stay}_{\mathsf{G}^{\cdot,\tau},\Phi}(E,s), \mathsf{exit}^{\Box}_f(E)). \tag{6}$$

Dually, for a Maximizer strategy σ and f lower bound, we get

$$\mathsf{V}_{\mathsf{G},\Phi}(s) \geq \min(\mathsf{stay}_{\mathsf{G}^{\sigma,\cdot},\Phi}(E,s), \mathsf{exit}^{\bigcirc}_f(E)). \tag{7}$$

Proof: We prove the Maximizer case, the Minimizer follows analogously. Since E is SEC, we know that all states in E obtain the same value in G by Lemma 2. Let τ^* an optimal Minimizer strategy. Then, all states of E also have the same optimal value in the induced MDP $\mathsf{G}^{\cdot,\tau^*}$, too.

Now, suppose an optimal Maximizer strategy σ^* remains in the EC, i.e. the probability to reach an state outside of E from s under (σ^*,τ^*) is 0. Then, we have $\mathsf{V}_{\mathsf{G},\Phi}(s) = \mathsf{stay}_{\mathsf{G}^{\cdot,\tau^*}}(E,s)$. Using the intuition of Eq. (4), we get $\mathsf{stay}_{\mathsf{G}^{\cdot,\tau^*}}(E,s) \leq \mathsf{stay}_{\mathsf{G}^{\cdot,\tau}}(E,s)$ for an arbitrary Minimizer strategy τ. For the second part, suppose σ^* leaves the EC with non-zero probability at some state $\hat{s}$. The value obtained in $\hat{s}$ by leaving is at most as large as the best exit $\mathsf{exit}^{\Box}_{\mathsf{V}_{\mathsf{G},\Phi}}(E)$ by definition of best exit, i.e. $\mathsf{V}_{\mathsf{G},\Phi}(\hat{s}) \leq \mathsf{exit}^{\Box}_{\mathsf{V}_{\mathsf{G},\Phi}}(E)$. Since f is an upper bound on the value, we also get $\mathsf{exit}^{\Box}_{\mathsf{V}_{\mathsf{G},\Phi}}(E) \leq \mathsf{exit}^{\Box}_f(E)$. Consequently, $\mathsf{V}_{\mathsf{G},\Phi}(\hat{s}) \leq \mathsf{exit}^{\Box}_f(E)$. Now, to transfer the result from the exiting state $\hat{s}$ to all other states, we crucially require the SEC assumption and, in particular, Corollary 1. This yields $\mathsf{V}_{\mathsf{G},\Phi}(s) = \mathsf{V}_{\mathsf{G},\Phi}(\hat{s})$.

Together, we get that $\mathsf{V}_{\mathsf{G},\Phi}(s) \leq \mathsf{stay}_{\mathsf{G}^{\cdot,\tau}}(E,s)$ in case an optimal Maximizer strategy remains inside the EC and $\mathsf{V}_{\mathsf{G},\Phi}(s) = \mathsf{V}_{\mathsf{G},\Phi}(\hat{s}) \leq \mathsf{exit}^{\Box}_f(E)$ if an optimal Maximizer strategy leaves. This proves the claim. ■

F. Proof of Theorem 3

Theorem 3. *Algorithm 2 is correct and terminates for any considered objective* Φ.

Proof: Correctness: We prove that $\mathsf{L}_i \leq \mathsf{V} \leq \mathsf{U}_i$ by induction. The bounds are updated in three places: (i) the initialization (Line 2), (ii) the updates (Lines 8 and 9), and (iii) the inflating and deflating (Lines 13 and 17).

The bounds are correct after initialization by assumption, see Appendix B. For the updates of the upper bounds (lower bounds follow analogously), observe that even updating with the optimal choice, i.e. setting

$$\mathsf{U}_i(s) = \mathsf{Fix}_\Phi(\mathsf{U}_{i-1}, s) = \\ v(s) + \mathsf{opt}_{a \in \mathsf{A}(s)} \sum\nolimits_{s' \in \mathsf{S}} \delta(s, a, s') \cdot \mathsf{U}_{i-1}(s') \geq \\ \mathsf{Fix}_\Phi(\mathsf{V}_{\mathsf{G},\Phi}),$$

is correct by linearity of the objective and the induction hypothesis $\mathsf{U}_{i-1} \geq \mathsf{V}_{\mathsf{G},\Phi}$. See also [7], [21] for objective-specific reasoning. Since we fix an arbitrary choice of the Minimizer for the update, $\mathsf{U}_i(s) \geq \mathsf{Fix}_\Phi(\mathsf{U}_{i-1}, s) \geq \mathsf{V}_{\mathsf{G},\Phi}(s)$ in all Minimizer states. For the correctness of deflating and inflating, observe that we can directly apply Lemma 5.

Together, we obtain correctness, i.e. $\mathsf{L}_i \leq \mathsf{V} \leq \mathsf{U}_i$ for all i.

Termination: Similar to the proof of Algorithm 1, this proof relies on the strategy recommender to eventually give optimal strategies (σ^*, τ^*). Moreover, for the local algorithm we require the strategy recommender to be stable. Note that the recommended strategies may change infinitely often, we only know that eventually each recommended strategy is optimal. Thus, the fixpoint updates as well as deflating and inflating steps eventually are only applied to "optimal MDPs" $\mathsf{G}^{\cdot,\tau^*}$ respectively $\mathsf{G}^{\sigma^*,\cdot}$.

It remains to show that the repeated updates of upper bounds together with deflating converge to the correct solution. The dual statement for inflating follows analogously. Since there are only finitely many memoryless deterministic strategies, there exists an optimal Minimizer strategy τ^* which is recommended infinitely often. Let k_i denote the sequence of occurrences of τ^*.

Since $\mathsf{U}_i(s)$ is monotonically decreasing and bounded from below by $\mathsf{V}_{\mathsf{G},\Phi}(s)$ (as shown in the correctness proof), it converges to some value $\mathsf{U}(s) = \lim_{i \to \infty} \mathsf{U}_i(s) \geq \mathsf{V}_{\mathsf{G},\Phi}(s)$. (Note that this limit is potentially never attained during the execution of the algorithm.) By definition of $\mathsf{U}_i(s)$, we necessarily have that $\mathsf{U}(s)$ is a fixpoint of $\mathsf{Fix}_\Phi^{\mathsf{G}^{\cdot,\tau^*}}$, i.e. Fix_Φ applied in the MDP. Let σ^* be the Maximizer strategy which takes all actions that are witnesses for this fixpoint uniformly at random, i.e. for each Maximizer state $s \in \mathsf{S}_\Box$ the strategy takes all actions $a \in \mathsf{A}(s)$ which achieve the optimal value $\mathsf{Fix}_\Phi(\mathsf{U}, s)$. Define the induced MC $\mathsf{M}^* = \mathsf{G}^{\sigma^*,\tau^*}$. By construction, U also is a fixpoint of $\mathsf{Fix}_\Phi^{\mathsf{M}^*}$. Fix an arbitrary BSCC B of M^*. This BSCC corresponds to an EC E in $\mathsf{G}^{\cdot,\tau^*}$, on which U is a fixpoint of Fix_Φ in $\mathsf{G}^{\cdot,\tau^*}$. Thus, E is SEC in $\mathsf{G}^{\cdot,\tau^*}$ for U and $\mathsf{U}(s) = \mathsf{U}(t)$ for all states $s, t \in E$. Moreover, in every state each exiting action has value *strictly* smaller than all staying action w.r.t. U (otherwise σ^* would choose this action, too), i.e. $\mathsf{exit}_{\mathsf{U}}^{\Box}(E) < \mathsf{U}(s)$ for all s. This also means that E is inclusion maximal: There is no action that could be added without violating the SEC condition. As U is the limit of U_i and the best exit of E under U is strictly smaller than the staying actions of E, there necessarily exists a step i where the same holds for the best exit under U_i. Moreover, since τ^* is optimal, we have that $\mathsf{stay}_{\mathsf{G}^{\cdot,\tau^*},\Phi}(E, s) \leq \mathsf{V}_{\mathsf{G},\Phi}(s)$. Since E is MSEC in the MDP $\mathsf{G}^{\cdot,\tau^*}$, it is repeatedly deflated by the algorithm. Together, if $\mathsf{V}_{\mathsf{G},\Phi}(s) < \mathsf{U}(s)$ for some state s in E, we in particular have that $\mathsf{U}_i(s) > \max(\mathsf{stay}_{\mathsf{G}^{\cdot,\tau^*},\Phi}(E, s), \mathsf{exit}_{\mathsf{U}_i}^{\Box}(E))$, contradicting deflation. In summary, we eventually have that $\mathsf{U}_i(s) = \mathsf{V}_{\mathsf{G},\Phi}(s)$ for all BSCCs of M^*.

It remains to argue the same about the other states. For every i, set σ_i the Maximizer strategy which takes all U_{k_i}-optimal actions uniformly at random. There are only finitely many different strategies σ_i (a strategy choosing actions uniformly at random is uniquely characterized by the set of actions it chooses in each state). Hence let $\hat{\sigma}$ now equal such a strategy that appears infinitely often in that sequence. The upper bounds U_i are updated infinitely often according to the transition structure of $\mathsf{G}^{\hat{\sigma},\tau^*}$ by choice of σ_i and $\hat{\sigma}$, i.e. $\mathsf{U}_{k'_i+1}(s) \leq \mathsf{Fix}_\Phi^{\mathsf{G}^{\hat{\sigma},\tau^*}}(\mathsf{U}_{k'_i}, s)$ where k'_i are the indices of occurrences of $\hat{\sigma}$. By applying a standard absorption argument, U necessarily is a fixpoint of $\mathsf{Fix}_\Phi^{\mathsf{G}^{\sigma_i,\tau^*}}$. Moreover, the BSCCs of $\mathsf{G}^{\hat{\sigma},\tau^*}$ necessarily all are subsets of BSCCs of M^*, by the previous argument the values on these BSCCs equal the true values, and by classical results on Markov chains we get convergence to the true value on all remaining states: By applying the same arguments as before (considering all paths of finite length k and letting k tend to infinity), we observe that for fixpoint-linear objectives there is a unique fixpoint on Markov chains once values on all BSCCs are fixed. Intuitively, since we eventually reach BSCCs with probability one, the unique solution of the fixpoint equations is given by the values on BSCCs, weighted by the probability of reaching them, together with the weighted sum of all transient paths (which is uniquely determined). Since we have a unique fixpoint, and value iteration is guaranteed to converge to that unique fixpoint for all objectives, we obtain the result. ■

APPENDIX D
STRATEGY ITERATION VARIANT OF ALGORITHM 1

Algorithm 3 shows how we can modify Algorithm 1 to work with strategy iteration as the strategy recommender. The key elements stay the same: in every iteration, the strategy recommender gives us a pair of strategies (Line 7) which are then used to update the under- and over-approximations (Lines 8 and 9).

Lemma 12. *Let* Φ *a reachability, safety, or mean payoff objective. Then Algorithm 3 is correct and terminates, i.e. for all* i*, we have* $\mathsf{L}_i \leq \mathsf{V}_{\mathsf{G},\Phi} \leq \mathsf{U}_i$*, and for every* $\varepsilon \geq 0$*, there is* i *with* $\mathsf{U}_i(s_0) - \mathsf{L}_i(s_0) \leq \varepsilon$.

Algorithm 3 Generic strategy iteration anytime algorithm

```
Input: SG $\mathsf{G}$, Objective $\Phi$, initial state $s_0$, and precision $\varepsilon$
Output: $(\mathsf{L}, \mathsf{U})$ such that $\mathsf{L} \leq \mathsf{V} \leq \mathsf{U}$ and $\mathsf{U}(s_0) - \mathsf{L}(s_0) \leq \varepsilon$
    # Initialization
 1: $\mathsf{L}_0(\cdot) \leftarrow 0$, $\mathsf{U}_0(\cdot) \leftarrow \infty$        # Lower and upper bounds
 2: $\sigma \leftarrow$ arbitrary Maximizer strategy
 3: $i \leftarrow 0$
 4: repeat
 5:     $i \leftarrow i + 1$

    # Recommender procedure
 6:     $x \leftarrow \mathsf{V}(\mathsf{G}^{\sigma,\cdot})$
 7:     $\sigma, \tau \leftarrow$ best-effort strategies inferred from $x$

    # Compute bounds
 8:     $\mathsf{L}_i \leftarrow x$
 9:     $\mathsf{U}_i \leftarrow \mathsf{V}(\mathsf{G}^{\cdot,\tau})$
10: until $\mathsf{U}_i(s_0) - \mathsf{L}_i(s_0) \leq \varepsilon$
```

Proof: The proof is analogous to the proof of Theorem 1. It only remains to show that strategy iteration yields a strategy recommender. The sequence of Maximizer strategies is monotonically improving, see e.g. [22] for reachability (and by duality safety) and [49] for mean payoff. Moreover, there are only finitely many strategies, and hence eventually σ is optimal and we compute the optimal strategy of Minimizer τ as the best response in Lines 6 and 7. Thus, strategy iteration is a stable strategy recommender. ■

We comment on several interesting details: Firstly, we deliberately exclude total reward, since to the best of our knowledge strategy iteration for total reward has not been considered so far.

Further, in Line 8 we reuse the x computed in Line 6. Moreover, observe that by solving the MDP in Line 6, we typically directly obtain the optimal counter-strategy to σ, i.e. τ. So when obtaining σ and τ from x, τ is exactly the globally optimal strategy in the MDP $\mathsf{G}^{\sigma,\cdot}$; in contrast, the updated Maximizer strategy σ is only locally optimal against τ. Trying to use a global best response to τ, i.e. solving the MDP $\mathsf{G}^{\cdot,\tau}$, can result in cyclic behaviour and non-convergence to the optimal strategies [5]. This is why the computation of U_i does not affect the choice of the next strategies at all, but only serves to provide an upper bound. This is also the only real difference to classic strategy iteration: We additional compute the upper bound in Line 9.

Next, we can also consider the dual algorithm, which computes the sequence of Minimizer strategies. Starting from an arbitrary Minimizer strategy τ, we compute x as $\mathsf{V}(\mathsf{G}^{\cdot,\tau})$ and thus get an upper bound U_i on the value. For the lower bound, we compute $\mathsf{V}(\mathsf{G}^{\sigma,\cdot})$ with the current counter-strategy σ of Maximizer.

Finally, since our algorithm only complements standard SI with bounds on the precision, it does not affect the complexity of SI. Thus, we know that it takes at most exponentially many iterations (as there are exponentially many MD strategies); and in the worst case, it can require exponentially many iterations [25]. Independently, if we do not require an exact solution, our anytime bounds can lead to earlier termination.

Appendix E
Computing the Staying Value

In this section, we describe how to compute the staying value. We group reachability, safety and total reward objectives together in Appendix E-A, because for all these objectives the staying value can be obtained by graph analysis. Appendix E-B describes the computation for mean payoff. Here, we differentiate precisely computing the staying value and approximating it. The latter is more practical, since we can interleave computation of the staying value with the overall computation, in the spirit of [14].

A. Reachability, Safety and Total Reward

In this subsection, we briefly describe how to compute the staying value for all objectives we consider except mean payoff. Recall that we compute the staying value for a given memoryless Minimizer strategy τ, i.e. we consider the MDP $\mathsf{G}^{\cdot,\tau}$ restricted to an EC E. (Or, dually, for a given Maximizer strategy.) As such, this is a well known problem for all three objectives. For reachability and safety, recall that we assumed target states to be absorbing, thus an EC either equals a target state or does not contain any at all. As such, the staying values in these cases trivially are 0 and 1 for any non-target EC. (Note that this logic can also be applied directly on G.)

For total reward, recall that we assumed no state has infinite value *in the game*. Yet, it may still be the case that against a sub-optimal strategy τ, Maximizer can actually achieve a value of infinity. However, the computation still is rather straightforward: Since E is an EC in $\mathsf{G}^{\cdot,\tau}$, we can reach any state infinitely often. Thus, if there exists a state with non-zero reward in E, we can definitely obtain infinite total reward. Otherwise, i.e. all states in E have $\mathsf{r}(s) = 0$, clearly the staying value is 0. In an MDP $\mathsf{G}^{\sigma,\cdot}$ where Minimizer is playing, an EC has infinite staying value if Minimizer cannot avoid visiting a state with non-zero reward infinitely often. This can be computed using standard graph analysis. Note that such an EC with infinite staying value cannot be simple, since by assumption Minimizer has a strategy that exits every EC containing a state with non-zero reward (already in the game, i.e. even against an optimal Maximizer strategy).

B. Mean Payoff

For mean payoff, the staying value of an EC E can have a non-extremal value, i.e. not 0, 1 or ∞. As before, the computation corresponds to solving the MDP $\mathsf{G}^{\cdot,\tau}_{|E}$.

If we want the precise value, we can obtain it by classical methods, such as linear programming or strategy iteration [31, Chp. 8 & 9]. However, solving all ECs precisely in each step is expensive; moreover, it is unnecessary when either (i) E was a wrong guess for a SEC or (ii) full precision in the EC is not needed to achieve convergence, e.g. because exiting it is optimal. We now describe how to modify Algorithm 2 to a more "on-demand", purely VI style, i.e. to rely on successive approximations of the staying values. This corresponds to the on-demand value iteration for mean payoff in MDP, see [14, Sec. 3.3]. We replace $\mathsf{stay}_{\mathsf{G}^{\cdot,\tau},\Phi}(E,s)$ with an over-approximation $\mathsf{Ustay}_{\mathsf{G}^{\cdot,\tau},\Phi}(E,s)$ in Line 12, and, dually, use $\mathsf{Lstay}_{\mathsf{G}^{\sigma,\cdot},\Phi}(E,s)$ in Line 16.

We compute the approximations as follows:

- *Successively more precise value iteration*: We build on an idea from [14, Sec. 3.3]. We use value iteration to compute the value in the MDP. This allows us to stop before reaching full precision. Concretely, for a given EC E in $\mathsf{G}^{\cdot,\tau}$ (or the dual for Minimizer), we have a precision ε' (independent of the ε of the overarching algorithm). We stop the local value iteration when precision ε' is reached and can infer lower and upper bounds on the value of the restricted MDP; a more detailed description of this process follows in the proof of correctness below. The next time we consider the same EC, we decrease ε', e.g. by halving it. On the one hand, this allows us to avoid spending time on ECs that were guessed wrongly, since we only solve them to precision ε'. On the other hand, since we have a convergent strategy recommender, from some point onward, all relevant ECs are found in every iteration of the main loop. Thus, the computation of staying values becomes more and more precise, eventually reaching the required precision.
- *Early stopping*: We improve the previous idea with the additional insight that the staying value is relevant if and only if both players do not want to leave the EC. Thus, if the upper bound inferred from the value iteration is smaller than Maximizer's best exit, or dually the lower bound larger than Minimizer's best exit, we can also stop the value iteration.
- *Initialization*: One could start the value iteration in the restricted MDP from a vector x_0 that assigns 0 to all states. However, the value iteration for mean payoff converges no matter which initial values are used. Thus, it is also correct and potentially a lot faster to start (i) from the x_i that the overarching algorithm has computed so far or (ii) from the approximations that a previous computation of the staying value for the given EC has computed.

To summarize, we compute approximations $\mathsf{Ustay}_{\mathsf{G}^{\cdot,\tau},\Phi}(E,s)$ and $\mathsf{Lstay}_{\mathsf{G}^{\sigma,\cdot},\Phi}(E,s)$ as follows: We run value iteration in the induced MDP restricted to the given end component, either until a required precision ε' is reached or until the staying value is clearly irrelevant. Also, we can re-use previous intermediate results, since the value iteration converges independently of the starting values. We argue that this modification preserves correctness and convergence.

Lemma 13. *Modify Algorithm 2 as described, i.e. using* $\mathsf{Ustay}_{\mathsf{G}^{\cdot,\tau},\Phi}(E,s)$ *in Line 12 and* $\mathsf{Lstay}_{\mathsf{G}^{\sigma,\cdot},\Phi}(E,s)$ *in Line 16; and computing these approximations of the staying value through successively more precise value iteration.*

This modified version is correct and terminates.

Proof: The proof is the mainly the same as the proof of Theorem 3. We only address the differences.

Correctness: Correctness follows immediately from known results of mean payoff computation on MDP, which we recall briefly. In particular, we restate some claims of [31, Sec. 8.5]: Running value iteration according to the total reward Bellman equation (Eq. (3)), we get a sequence of vectors $(x_i)_{i\in\mathbb{N}}$. Let $\Delta_i := x_{i+1} - x_i$ be the difference vector, i.e. the reward gained in the $i+1$-th step. We can show that[(5)]

$$\min_{s\in\mathsf{S}} \Delta_i(s) \leq v \leq \max_{s\in\mathsf{S}} \Delta_i(s). \tag{8}$$

Thus, when $\max_{s\in\mathsf{S}} \Delta_i(s) - \min_{s\in\mathsf{S}} \Delta_i(s) \leq \varepsilon'$, we know that we are ε'-close to the value v and can stop value iteration. Moreover, $\min_{s\in\mathsf{S}} \Delta_i(s)$ is a lower bound on v and $\max_{s\in\mathsf{S}} \Delta_i(s)$ an upper bound on v. In conclusion: Running value iteration as in [31, Sec. 8.5] until ε' precision is reached, we can return valid lower respectively upper bounds on the staying value in the given MDP.

We address the two additional differences: For our "early stopping", observe that we take the $\max$ in Line 12 and the $\min$ in Line 16. This means that we stop computation early if and only if our approximation of the staying value certainly is smaller respectively larger than the exit value we compare to, meaning that further tightening the precision of the result has no influence on the algorithm anyway. Finally, for "restarting", note that Eq. (8) is independent of the starting vector.

Termination: Since we have a convergent strategy recommender, eventually we always investigate the same ECs, namely those that are ECs under optimal strategies. Note that, as the strategies are optimal, all states of the EC have the same value *in the SG*, and thus the ECs are SECs. Every time we approximate the staying value, we decrease the precision, and the limit of the sequence of our precisions is 0. The value iteration for mean payoff on the MDP is guaranteed to terminate for any precision requirement, thus, eventually the precision of the staying value is small enough that the overarching algorithm can terminate based on the computed staying values. ■

[(5)]Technically, we also need that the MDP is *aperiodic*. For this, we employ a standard aperiodicity transformation from [31, Sec. 8.5.4].