Life Is Random, Time Is Not: Markov Decision Processes with Window Objectives

Thomas Brihaye
UMONS – Université de Mons, Belgium

Florent Delgrange
UMONS – Université de Mons, Belgium
RWTH Aachen, Germany

Youssouf Oualhadj
LACL – UPEC, Paris, France

Mickael Randour
F.R.S.-FNRS & UMONS – Université de Mons, Belgium

Abstract

The window mechanism was introduced by Chatterjee et al. [17] to strengthen classical game objectives with time bounds. It permits to synthesize system controllers that exhibit acceptable behaviors within a configurable time frame, all along their infinite execution, in contrast to the traditional objectives that only require correctness of behaviors in the limit. The window concept has proved its interest in a variety of two-player zero-sum games, thanks to the ability to reason about such time bounds in system specifications, but also the increased tractability that it usually yields. In this work, we extend the window framework to stochastic environments by considering the fundamental *threshold probability problem* in Markov decision processes for window objectives. That is, given such an objective, we want to synthesize strategies that guarantee satisfying runs with a given probability. We solve this problem for the usual variants of window objectives, where either the time frame is set as a parameter, or we ask if such a time frame exists. We develop a generic approach for window-based objectives and instantiate it for the classical mean-payoff and parity objectives, already considered in games. Our work paves the way to a wide use of the window mechanism in stochastic models.

2012 ACM Subject Classification Software and its engineering → Formal methods; Theory of computation → Logic and verification; Theory of computation → Markov decision processes

Keywords and phrases Markov decision processes, window mean-payoff, window parity

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2019.8

Related Version Full version available at `https://arxiv.org/abs/1901.03571`.

Funding Research supported by F.R.S.-FNRS, Grant n° F.4520.18 (ManySynth), and F.R.S.-FNRS mobility funding for scientific missions (Y. Oualhadj in UMONS, 2018).
Mickael Randour: F.R.S.-FNRS Research Associate.

1 Introduction

Game-based models for controller synthesis. *Two-player zero-sum games* [28, 35] and *Markov decision processes* (MDPs) [26, 4, 36] are two popular frameworks to model decision making in adversarial and uncertain environments respectively. In the former, a system controller and its environment compete antagonistically, and synthesis aims at building strategies for the controller ensuring a specified behavior *against all possible strategies of the environment*. In the latter, the system is faced with a given stochastic model of its environment, and the focus is on satisfying a given level of expected performance, or a

specified behavior with a sufficient probability. Classical objectives studied in both settings include *parity*, a canonical way of encoding ω-regular specifications, and *mean-payoff*, which evaluates the average payoff per transition in the limit of an infinite run in a weighted graph.

Window objectives in games. The traditional parity and mean-payoff objectives share two shortcomings. First, they both reason about infinite runs *in their limit*. While this elegant abstraction yields interesting theoretical properties and makes for robust interpretation, it is often beneficial in practical applications to be able to specify a *parameterized time frame* in which an acceptable behavior should be witnessed. Second, both parity and mean-payoff games belong to $\mathsf{UP} \cap \mathsf{coUP}$ [34, 27], but despite recent breakthroughs [16, 22], they are still not known to be in P. Furthermore, the latest results [21, 25] indicate that all existing algorithmic approaches share inherent limitations that prevent inclusion in P.

Window objectives address the time frame issue as follows. In their *fixed* variant, they consider a window of size bounded by $\lambda \in \mathbb{N}_0$ (given as a parameter) sliding over an infinite run and declare this run to be winning if, in all positions, the window is such that the (mean-payoff or parity) objective is locally satisfied. In their *bounded* variant, the window size is not fixed a priori, but a run is winning if there exists a bound λ for which the condition holds. Window objectives have been considered both in *direct* versions, where the window property must hold from the start of the run, and *prefix-independent* versions, where it must hold from some point on. Window games were initially studied for mean-payoff [17] and parity [14]. They have since seen diverse extensions and applications: e.g., [5, 3, 11, 15, 32, 38].

Window objectives in MDPs. Our goal is to lift the theory of window games to the stochastic context. With that in mind, we consider the canonical *threshold probability problem*: given an MDP, a window objective defining a set of acceptable runs E, and a probability threshold α, we want to decide if there exists a controller *strategy* (also called *policy*) to achieve E with probability at least α. It is well-known that many problems in MDPs can be reduced to threshold problems for appropriate objectives: e.g., maximizing the *expectation* of a prefix-independent function boils down to maximizing the probability to reach the best end-components for that function (see examples in [4, 13, 37]).

Example. Before going further, let us consider an example. Take the MDP depicted in Fig. 1(a): circles depict states and dots depict actions, labeled by letters. Each action yields a probability distribution over successor states: for example, action b leads to s_2 with probability 0.5 and s_3 with the same probability. This MDP is actually a *Markov chain* (MC) as the controller has only one action available in each state: this process is purely stochastic.

We consider the *parity* objective here: each state is associated with a non-negative integer priority and a run is winning if the minimum one amongst those seen infinitely often is even. Clearly, any run in this MC is winning: either it goes through s_3 infinitely often and the min. priority is 0, or it does not, and the min. priority seen infinitely often is 2. Hence the controller not only wins *almost-surely* (with probability one), but even *surely* (on all runs).

Now, consider the *window parity* objective that informally asks for the minimum priority inside a window of size bounded by λ to be even, with this window sliding all along the infinite run. Fix any $\lambda \in \mathbb{N}_0$. It is clear that every time s_1 is visited, there will be a fixed strictly positive probability $\varepsilon > 0$ of not seeing 0 before λ steps: this probability is $1/2^{\lambda-1}$. Let us call this seeing a *bad window*. Since we are in a *bottom strongly connected component* of the MC, we will almost-surely visit s_1 infinitely often [4]. Using classical probability arguments (Borel-Cantelli), one can easily be convinced that the probability to see bad

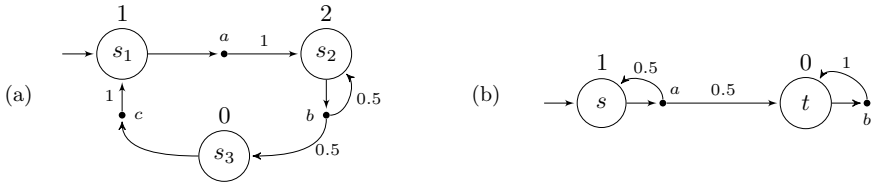

Figure 1 Markov chains where (a) parity is surely satisfied but all window parity objectives have probability zero, and (b) there is no uniform bound over all runs, in contrast to the game setting.

windows infinitely often is one. Hence the probability to win the window parity objective is zero. This canonical example illustrates the difference between traditional parity and window parity: the latter is more restrictive as it asks for a strict bound on the time frame in which each odd priority should be answered by a smaller even priority.

Note that in practice, such a behavior is often wished for. For example, consider a computer server having to grant requests to clients. A classical parity objective can encode that requests should eventually be granted. However, it is clear that in a desired controller, requests should not be placed on hold for an arbitrarily long time. The existence of a finite bound on this holding time can be modeled with a *bounded* window parity objective, while a specific bound can also be set as a parameter using a *fixed* window parity objective.

Our contributions. We study the *threshold probability problem* in MDPs for window objectives based on *parity* and *mean-payoff*, two prominent formalisms in qualitative and quantitative (resp.) analysis of systems. We consider the different variants of window objectives mentioned above: fixed vs. bounded, direct vs. prefix-independent. A nice feature of our approach is that we provide a *unified view* of parity and mean-payoff window objectives: *our algorithm can actually be adapted for any window-based objective* if an appropriate black-box is provided for a restricted sub-problem. This has two advantages: (i) conceptually, our approach permits a *deeper understanding of the essence of the window mechanism*, not biased by technicalities of the specific underlying objective; (ii) our framework can *easily* be *extended* to other objectives for which a window version could be defined. This point is of great practical interest too, as it opens the perspective of a modular, generic software tool suite for window objectives.

For the sake of space, we use acronyms below: DFW for direct fixed window, FW for (prefix-independent) fixed window, DBW for direct bounded window, and BW for (prefix-independent) bounded window. Our main contributions are as follows.

1. We solve DFW MDPs through reductions to safety MDPs over *well-chosen unfoldings*. This results in polynomial-time and pseudo-polynomial-time algorithms for the parity and mean-payoff variants respectively. We prove these complexities to be almost tight (Thm. 5), the most interesting case being the PSPACE-hardness of DFW mean-payoff objectives, even in the case of *acyclic* MDPs.
   We also show that no upper bound can be established on the window size needed to win in general (Ex. 3), *in stark contrast to the two-player games situation* (Rmk. 2).
2. We use similar reductions to prove that *finite memory* suffices in the prefix-independent case (Thm. 4). In this case, we can do better than using unfoldings to solve the problem. We study *end-components* (ECs), the crux for all prefix-independent objectives in MDPs: we show that ECs can be classified based on their *two-player zero-sum game interpretation* (Sect. 5). Using the result on finite memory, we prove that in ECs classified as *good*, almost-sure satisfaction of window objectives can be ensured, whereas it is impossible to

satisfy them with non-zero probability in other ECs (Lem. 10). We also establish tight complexity bounds for this classification problem (Thm. 15). This *EC classification* is (conceptually and complexity-wise) the cornerstone to deal with general MDPs.

3. Our general algorithm is developed in Sect. 6: we prove P-completeness for all prefix-independent variants but for the BW mean-payoff one (Thm. 17), where we show that the problem is in $\mathsf{NP} \cap \mathsf{coNP}$ and as hard as mean-payoff games, a canonical "hard" problem for that complexity class.
4. For all variants, we prove *tight memory bounds*: see Thm. 5 for the direct fixed variants, Thm. 17 for the prefix-independent fixed and bounded ones. In all cases, *pure* strategies (i.e., without randomness) suffice.
5. We leave out DBW objectives from our analysis as we show they are not well-behaved. We illustrate their behavior in Sect. 3 and discuss their pitfalls in Sect. 7.

Along the way, we develop side results that help drawing a *line between MDPs and games* w.r.t. window objectives: e.g., the existence of a uniform bound in the bounded case (Rmk. 2).

In the game setting, window objectives are all in polynomial time, except for the BW mean-payoff variant, in $\mathsf{NP} \cap \mathsf{coNP}$. Despite clear differences in behaviors, the situation is almost the same here. The only outlier case is the DFW mean-payoff one, whose complexity rises significantly (Thm. 5): the loss of prefix-independence permits to emulate shortest path problems on MDPs, famously hard to solve efficiently (e.g., [30, 37, 9, 31]). In the almost-sure case, however, DFW mean-payoff MDPs collapse to P (Rmk. 6).

In games, window objectives permit to avoid long-standing $\mathsf{NP} \cap \mathsf{coNP}$ complexity barriers for parity [14] and mean-payoff [17]. Since both are known to be in P for the threshold probability problem in MDPs [20, 37], the main interest of window objectives resides in their *modeling power*. Still, they may turn out to be more efficient in practice too, as polynomial-time algorithms for parity and mean-payoff, based on linear programming, are often forsaken in favor of exponential-time value or strategy iteration ones (e.g., [2]).

Related work. We already mentioned many related articles, hence we only briefly discuss some remaining models here. Window parity games are strongly linked to the concept of *finitary ω-regular games*: see, e.g., [19], or [14] for a complete list of references. The window mechanism can be used to ensure a certain form of (local) guarantee over runs: different techniques have been considered in MDPs, notably *variance-based* [10] or *worst-case-based* [13, 7] methods.

Finally, let us mention the recent work of Bordais et al. [8], considering a *seemingly* related question: the authors define a *value function* based on the window *mean-payoff* mechanism and consider maximizing its expected value (which is different from the expected window size we discuss in Sect. 7). While there are similarities in our works w.r.t. technical tools, the two approaches are quite different and have their own strengths: we focus on deep understanding of the window mechanism through a *generic approach* for the canonical threshold probability problem for all window-based objectives, here instantiated as mean-payoff *and* parity; whereas Bordais et al. focus on a particular *optimization problem* for a function relying on the window mechanism. We mention three examples illustrating the conceptual gap. First, in [8], the studied function takes the same value for direct and prefix-independent bounded window mean-payoff objectives, whereas we show in Sect. 3 that the classical definitions of window objectives induce a striking difference between both (in the MDP of Ex. 3, the prefix-independent version is satisfied for window size one, whereas no uniform bound on all runs can be defined for the direct case). Second, we prove PSPACE-hardness for the DFW mean-payoff case, whereas the best lower bound known for the related problem in [8] is PP. Lastly, recall that we also deal with window *parity* objectives while the function of [8] is strictly built on mean-payoff.

Outline. Sect. 2 defines the problem under study. In Sect. 3, we introduce window objectives, discuss their status in games, and illustrate their behavior in MDPs. Sect. 4 is devoted to the fixed variants and the aforementioned reductions. In Sect. 5, we analyze the case of ECs and develop the classification procedure. We build on it in Sect. 6 to solve the general case. Finally, in Sect. 7, we discuss the limitations of our work, as well as interesting extensions within arm's reach (e.g., multi-objective threshold problem, expected value problem). Full details and proofs can be found in the full version of this paper [12].

2 Preliminaries

Markov decision processes. Given a set S, let $\mathcal{D}(S)$ be the set of rational probability distributions over S. Given $\iota \in \mathcal{D}(S)$, let $\mathsf{Supp}(\iota) = \{s \in S \mid \iota(s) > 0\}$ be its support. A finite *Markov decision process* (MDP) is a tuple $\mathcal{M} = (S, A, \delta)$ where S is a finite set of *states*, A is a finite set of *actions* and $\delta \colon S \times A \to \mathcal{D}(S)$ is a partial function called the *probabilistic transition function*. The set of actions available in $s \in S$ (i.e., for which $\delta(s, a)$ is defined) is $A(s)$. We assume w.l.o.g. that MDPs are *deadlock-free*: for all $s \in S$, $A(s) \neq \emptyset$. An MDP where for all $s \in S$, $|A(s)| = 1$ is a fully-stochastic process called a *Markov chain* (MC).

A *run* of $\mathcal{M}$ is an infinite sequence $\rho = s_0 a_0 \dots a_{n-1} s_n \dots$ of states and actions such that $\delta(s_i, a_i, s_{i+1}) > 0$ for all $i \geq 0$. The *prefix* up to the n-th state of ρ is the finite sequence $\rho[0, n] = s_0 a_0 \dots a_{n-1} s_n$. The *suffix* of ρ starting from the n-th state of ρ is the run $\rho[n, \infty] = s_n a_n s_{n+1} a_{n+1} \dots$. Moreover, we denote by $\rho[n]$ the n-th state s_n of ρ. Finite prefixes of runs of the form $h = s_0 a_0 \dots a_{n-1} s_n$ are called *histories*. We resp. denote the sets of runs and histories of an MDP $\mathcal{M}$ by $\mathsf{Runs}(\mathcal{M})$ and $\mathsf{Hists}(\mathcal{M})$.

Strategies, induced MC and events. A *strategy* σ is a function $\mathsf{Hists}(\mathcal{M}) \to \mathcal{D}(A)$ such that for all $h \in \mathsf{Hists}(\mathcal{M})$ ending in s, we have $\mathsf{Supp}(\sigma(h)) \subseteq A(s)$. The set of all strategies is Σ. A strategy is *pure* if all histories are mapped to *Dirac distributions*, i.e., the support is a singleton. A strategy σ can be encoded by a stochastic state machine with outputs, called *Mealy machine*. We say that σ is *finite-memory* if this machine is finite, and memoryless if it has only one state, i.e., it only depends on the last state of the history. We see such strategies as functions $s \mapsto \mathcal{D}(A(s))$ for $s \in S$. The entity choosing the strategy is called the *controller*.

An MDP $\mathcal{M}$, a strategy σ, and a state s determine a Markov chain $\mathcal{M}_s^\sigma$. When considering the probabilities of events in $\mathcal{M}_s^\sigma$, we will often consider sets of runs of $\mathcal{M}$. Thus, given $E \subseteq (SA)^\omega$, we denote by $\mathbb{P}^\sigma_{\mathcal{M},s}[E]$ the probability of the runs of $\mathcal{M}_s^\sigma$ whose projection to $\mathcal{M}$ is in E, i.e., the probability of event E when $\mathcal{M}$ is executed with initial state s and strategy σ. For the sake of readability, we make similar abuse of notation – identifying runs in the induced MC with their projections in the MDP – throughout our paper.

Note that every measurable set (*event*) has a uniquely defined probability [40] (Carathéodory's extension theorem induces a unique probability measure on the Borel σ-algebra over cylinders of $(SA)^\omega$). When non-ambiguous, we drop some subscripts of $\mathbb{P}^\sigma_{\mathcal{M},s}$.

We say that an event $E \subseteq (SA)^\omega$ is *sure*, written $\mathsf{S}^\sigma_{\mathcal{M},s}[E]$, if and only if $\mathsf{Runs}(\mathcal{M}_s^\sigma) \subseteq E$ and that E is *almost-sure*, written $\mathsf{AS}^\sigma_{\mathcal{M},s}[E]$, if and only if $\mathbb{P}^\sigma_{\mathcal{M},s}[E] = 1$.

Limiting behavior. Fix an MDP $\mathcal{M} = (S, A, \delta)$. A *sub-MDP* of $\mathcal{M}$ is an MDP $\mathcal{M}' = (S', A', \delta')$ with $S' \subseteq S$, $\emptyset \neq A'(s) \subseteq A(s)$ for all $s \in S'$, $\mathsf{Supp}(\delta(s, a)) \subseteq S'$ for all $s \in S', a \in A'(s)$, $\delta' = \delta|_{S' \times A'}$. Such a sub-MDP $\mathcal{M}'$ is an *end-component* (EC) of $\mathcal{M}$ if and only if the underlying graph of $\mathcal{M}'$ is *strongly connected*, i.e., there is a run between any pair of states in S'. Given an EC $\mathcal{M}' = (S', A', \delta')$ of $\mathcal{M}$, we say that its sub-MDP $\mathcal{M}'' = (S'', A'', \delta'')$,

$S'' \subseteq S'$, $A'' \subseteq A'$, is a *sub-EC* of $\mathcal{M}'$ if $\mathcal{M}''$ is also an EC. We let $\mathsf{EC}(\mathcal{M})$ denote the set of ECs of $\mathcal{M}$, which may be of exponential size as ECs need not be disjoint. The counterparts of ECs in MCs are *bottom strongly-connected components* (BSCCs).

The union of two ECs with non-empty intersection is an EC: hence we can define the *maximal* ECs (MECs) of an MDP, i.e., the ECs that cannot be extended. We let $\mathsf{MEC}(\mathcal{M})$ denote the set of MECs of $\mathcal{M}$, of polynomial size (because MECs are pair-wise disjoint) and computable in polynomial time [18].

Given some run $\rho = s_0 a_0 s_1 a_1 \ldots \in \mathsf{Runs}(\mathcal{M})$, let $\mathsf{inf}(\rho) = \{s \in S \mid \forall\, i \geq 0, \exists\, j > i,\ s_j = s\}$ denote the states visited infinitely-often along ρ, and let $\mathsf{infAct}(\rho) = \{a \in A \mid \forall\, i \geq 0, \exists\, j > i,\ a_j = a\}$ similarly denote the actions taken infinitely-often along ρ. Let $\mathsf{limitSet}(\rho)$ denote the pair $(\mathsf{inf}(\rho), \mathsf{infAct}(\rho))$. Note that this pair may induce a well-defined sub-MDP $\mathcal{M}' = (\mathsf{inf}(\rho), \mathsf{infAct}(\rho), \delta|_{\mathsf{inf}(\rho)\times\mathsf{infAct}(\rho)})$, but in general it need not be the case. A folklore result in MDPs (e.g., [4]) is the following: for any state s of MDP $\mathcal{M}$, for any strategy $\sigma \in \Sigma$, we have that $\mathsf{AS}^{\sigma}_{\mathcal{M},s}[\{\rho \in \mathsf{Runs}(\mathcal{M}^{\sigma}_{s}) \mid \mathsf{limitSet}(\rho) \in \mathsf{EC}(\mathcal{M})\}]$, that is, under any strategy, the limit behavior of the MDP almost-surely coincides with an EC. This property is a key tool in the analysis of MDPs with prefix-independent objectives, as it essentially says that we only need to identify the "best" ECs and maximize the probability to reach them.

Weights, priorities and complexity. In this paper, we always assume an MDP with either (i) a *weight* function $w\colon A \to \mathbb{Z}$ of largest absolute weight W, or (ii) a *priority* function $p\colon S \to \{0, 1, \ldots, d\}$, with $d \leq |S| + 1$ (w.l.o.g.). This choice is left implicit when the context is clear, to offer a unified view of mean-payoff and parity variants of window objectives.

Regarding complexity, we make the classical assumptions of the field: we consider the model size $|\mathcal{M}|$ to be polynomial in $|S|$ and the *binary encoding* of weights and probabilities (e.g., $V = \log_2 W$), whereas we consider the largest priority d, as well as the upcoming window size λ, to be encoded in unary. When a problem is polynomial in W, we say that it is *pseudo*-polynomial: it would be polynomial if weights would be given in unary.

Objectives. An *objective* for an MDP $\mathcal{M} = (S, A, \delta)$ is a measurable set of runs $E \subseteq (SA)^{\omega}$. We consider window objectives based on *mean-payoff* and *parity* objectives. Let us discuss those classical objectives.

- *Mean-Payoff* is a *quantitative* objective for which we consider *weighted* MDPs. Let $\rho \in \mathsf{Runs}(\mathcal{M})$ be a run of such an MDP. The *mean-payoff* of prefix $\rho[0, n]$ is $\mathsf{MP}(\rho[0, n]) = \frac{1}{n}\sum_{i=0}^{n-1} w(a_i)$, for $n > 0$. This is naturally extended to runs by considering the limit behavior. The *mean-payoff* of ρ is $\mathsf{MP}(\rho) = \liminf_{n\to\infty} \mathsf{MP}(\rho[0, n])$. Given a threshold $\nu \in \mathbb{Q}$, the mean-payoff objective accepts all runs whose mean-payoff is above the threshold, i.e., $\mathsf{MeanPayoff}(\nu) = \{\rho \in \mathsf{Runs}(\mathcal{M}) \mid \mathsf{MP}(\rho) \geq \nu\}$. Note that ν can be taken equal to zero w.l.o.g., and the mean-payoff function can be equivalently (in the classical one-dimension setting) defined using lim sup.
- *Parity* is a *qualitative* objective for which we consider MDPs with a priority function. It requires that the smallest priority seen infinitely often along a run be even, i.e., $\mathsf{Parity} = \{\rho \in \mathsf{Runs}(\mathcal{M}) \mid \min_{s\in\mathsf{inf}(\rho)} p(s) = 0 \pmod 2\}$.

Decision problem. Given an MDP $\mathcal{M} = (S, A, \delta)$, an initial state s, a threshold $\alpha \in [0, 1]\cap\mathbb{Q}$, and an objective E, the *threshold probability problem* is to decide whether there exists a strategy $\sigma \in \Sigma$ such that $\mathbb{P}^{\sigma}_{\mathcal{M},s}[E] \geq \alpha$ or not.

Furthermore, if it exists, we want to build such a strategy. The related problems for both mean-payoff and parity are in P and pure memoryless strategies suffice [37, 20].

3 Window objectives

Good Windows. Given a weighted MDP $\mathcal{M}$ and $\lambda > 0$, we define the *good window mean-payoff* objective $\mathsf{GW}_{\mathsf{mp}}(\lambda) = \{\rho \in \mathsf{Runs}(\mathcal{M}) \mid \exists\, l < \lambda,\ \mathsf{MP}(\rho[0, l+1]) \geq 0\}$, requiring the existence of a window of size bounded by λ and starting at the first position of the run, over which the mean-payoff is at least equal to zero (w.l.o.g.).

Similarly, given an MDP $\mathcal{M}$ with priority function p, we define the *good window parity* objective, $\mathsf{GW}_{\mathsf{par}}(\lambda) = \{\rho \in \mathsf{Runs}(\mathcal{M}) \mid \exists\, l < \lambda,\ (p(\rho[l]) \bmod 2 = 0 \wedge \forall\, k < l,\ p(\rho[l]) < p(\rho[k]))\}$, requiring the existence of a window of size bounded by λ and starting at the first position of the run, for which the last priority is even and is the smallest within the window.

We use subscripts mp and par for mean-payoff and parity variants respectively. So, given $\Omega = \{\mathsf{mp}, \mathsf{par}\}$ and a run $\rho \in \mathsf{Runs}(\mathcal{M})$, we say that *an Ω-window is closed* in at most λ steps from $\rho[i]$ if $\rho[i, \infty]$ is in $\mathsf{GW}_\Omega(\lambda)$. If a window is not yet closed, we call it *open*.

Fixed variants. Given $\lambda > 0$, we define the *direct fixed window* objective $\mathsf{DFW}_\Omega(\lambda) = \{\rho \in \mathsf{Runs}(\mathcal{M}) \mid \forall\, j \geq 0,\ \rho[j, \infty] \in \mathsf{GW}_\Omega(\lambda)\}$, asking for all Ω-windows to be closed within λ steps along the run. We also define the *fixed window* objective $\mathsf{FW}_\Omega(\lambda) = \{\rho \in \mathsf{Runs}(\mathcal{M}) \mid \exists\, i \geq 0,\ \rho[i, \infty] \in \mathsf{DFW}_\Omega(\lambda)\}$, that is the *prefix-independent* version of the previous one: it requires it to be eventually satisfied.

Bounded variant. The *bounded window* objective $\mathsf{BW}_\Omega = \{\rho \in \mathsf{Runs}(\mathcal{M}) \mid \exists\, \lambda > 0,\ \rho \in \mathsf{FW}_\Omega(\lambda)\}$, requires the existence of a λ for which the fixed window objective is satisfied. Note that this bound need not be *uniform* along all runs in general. A *direct* variant may also be defined, but turns out to be ill-suited in the stochastic context: we illustrate it in Ex. 3 and discuss its pitfalls in Sect. 7. Hence we focus on the prefix-independent version here.

▶ **Example 1.** We first go back to the example of Sect. 1, depicted in Fig. 1(a). Let $\mathcal{M}$ be this MDP. Fix run $\rho = (s_1\, a\, s_2\, b\, s_3\, c)^\omega$. We have that $\rho \notin \mathsf{FW}_{\mathsf{par}}(\lambda = 2)$ – a fortiori, $\rho \notin \mathsf{DFW}_{\mathsf{par}}(\lambda = 2)$ – as the window that opens in s_1 is not closed after two steps (because s_1 has odd priority 1, and 2 is not smaller than 1 so does not suffice to answer it). If we now set $\lambda = 3$, we see that this window closes on time, as 0 is encountered within three steps. As all other windows are immediately closed, we have $\rho \in \mathsf{DFW}_{\mathsf{par}}(\lambda = 3)$ – a fortiori, $\rho \in \mathsf{FW}_{\mathsf{par}}(\lambda = 3)$ and $\rho \in \mathsf{BW}_{\mathsf{par}}$.

Regarding the probability of these objectives, however, we have already argued that, for all $\lambda > 0$, $\mathbb{P}_{\mathcal{M},s_1}[\mathsf{FW}_{\mathsf{par}}(\lambda)] = 0$, whereas $\mathbb{P}_{\mathcal{M},s_1}[\mathsf{Parity}] = 1$ since s_3 is almost-surely visited infinitely often but any time bound is almost-surely exceeded infinitely often too. Observe that $\mathsf{BW}_{\mathsf{par}} = \bigcup_{\lambda>0} \mathsf{FW}_{\mathsf{par}}(\lambda)$, hence we also have that $\mathbb{P}_{\mathcal{M},s_1}[\mathsf{BW}_{\mathsf{par}}] = 0$ (by countable additivity). Similar reasoning holds for window mean-payoff, by taking the weight function $w = \{a \mapsto -1,\ b \mapsto 0, c \mapsto 1\}$.

▶ Remark 2. Window mean-payoff and window parity objectives were considered in two-player zero-sum games [17, 14]. This setting is equivalent to deciding if there exists a strategy in an MDP such that the objective is *surely* satisfied, i.e., by all consistent runs. Interestingly, in games, if the controller can win the bounded window objective, then a uniform bound exists, i.e., there exists a window size λ sufficiently large such that the bounded version coincides with the fixed one. Recall that this is not granted by definition. This uniform bound is pseudo-polynomial for mean-payoff and equal to the number of states for parity. We illustrate in the following example that *in MDPs, a uniform bound need not exist.*

▶ **Example 3.** Consider the MC $\mathcal{M}$ in Fig. 1(b). For any $\lambda > 0$, there is probability $1/2^{\lambda-1}$ that objective $\mathsf{DFW}_{\mathsf{par}}(\lambda)$ is not satisfied. Hence, for all $\lambda > 0$, $\mathbb{P}_{\mathcal{M},s}[\mathsf{DFW}_{\mathsf{par}}(\lambda)] < 1$.

Now let $\mathsf{DBW}_{\mathsf{par}} = \bigcup_{\lambda>0} \mathsf{DFW}_{\mathsf{par}}(\lambda)$ be the *direct* bounded window objective evoked above. We claim that $\mathbb{P}_{\mathcal{M},s}[\mathsf{DBW}_{\mathsf{par}}] = 1$. Indeed, any run ending in t belongs to $\mathsf{DBW}_{\mathsf{par}}$, as it belongs to $\mathsf{DFW}_{\mathsf{par}}(\lambda)$ for λ equal to the length of the prefix up to t. Since t is almost-surely reached (as it is the only BSCC of the MC), we conclude that $\mathsf{DBW}_{\mathsf{par}}$ is indeed satisfied almost-surely.

Essentially, the difference stems from the fairness of probabilities. In a game, the opponent controls the successor choice after action a and always goes back to s, resulting in objective $\mathsf{DBW}_{\mathsf{par}}$ being lost. However, in this MC, s is almost-surely left eventually, but we cannot guarantee when: hence there exists a window bound for each run, but there is no uniform bound over all runs.

This MC also illustrates the difference between sure and almost-sure satisfaction: we have $\mathsf{AS}_{\mathcal{M},s}[\mathsf{FW}_{\mathsf{par}}(\lambda = 1)]$ but not $\mathsf{S}_{\mathcal{M},s}[\mathsf{FW}_{\mathsf{par}}(\lambda = 1)]$, because of run $\rho = (s\,a)^\omega$. Again the same reasoning holds for mean-payoff variants, for example with $w = \{a \mapsto -1, b \mapsto 1\}$.

We leave out the *direct* bounded objective from now on. We will come back to it in Sect. 7 and motivate why this objective is not well-behaved. In the following, we focus on direct and prefix-independent fixed window objectives and prefix-independent bounded ones.

4 Fixed case: better safe than sorry

We start with the fixed variants of window objectives. Our main goal here is to establish that *pure finite-memory* strategies suffice in all cases. As a by-product, we also obtain algorithms to solve the corresponding decision problems. Still, for the prefix-independent variants, we will obtain better complexities using the upcoming generic approach (Sect. 6).

Our tools are natural reductions from direct (resp. prefix-independent) window problems on MDPs to safety (resp. co-Büchi) problems on *unfoldings* based on the window size λ (i.e., larger arenas incorporating information on open windows).

Unfoldings. We use *identical unfoldings* for both direct and prefix-independent objectives. Let $\mathcal{M}$ be an MDP and $\lambda > 0$ be the window size. We build a new MDP $\mathcal{M}_\lambda$, the unfolding of $\mathcal{M}$ for mean-payoff (resp. parity), with an extended state space $\tilde{S}$: each state of $\mathcal{M}_\lambda$ is of the form $\tilde{s} = (s, l, x)$ so that s keeps track of the current state of $\mathcal{M}$, l of the size of the current open window, and x of the current sum of weights (resp. the minimum priority) in the window: the last two values are therefore reset whenever a window is closed or stays open for λ steps. The remaining components of $\mathcal{M}_\lambda$ are then extended from $\mathcal{M}$ in a natural way.

A key underlying property used here is the so-called *inductive property of windows* [17, 14]: for all runs $\rho = s_0 a_0 s_1 a_1 \dots$ of $\mathcal{M}$, fix a window starting in position $i \geq 0$ and let j be the position in which this window gets closed, assuming it does. Then, all windows in positions from i to j also close in j. The validity of this property is easy to check by contradiction (if it would not hold, the window in i would close before j). This property is fundamental in our reduction: without it we would have to keep track of all open windows in parallel, which would result in a blow-up exponential in λ.

Reductions. A safety (resp. co-Büchi) objective consists of runs avoiding at all times (resp. eventually avoiding) a given set of states B. In $\mathcal{M}_\lambda$, B is composed of states (s, l, x) where $l = \lambda$ and $x < 0$ for the mean-payoff variant or $l = \lambda - 1$ and $x \bmod 2 = 1$ for the parity variant, that exactly correspond to windows staying open for λ steps.

We state that the safety and co-Büchi objectives in $\mathcal{M}_\lambda$ are *probability-wise equivalent* to the direct fixed window and fixed window ones in $\mathcal{M}$. For any strategy σ in $\mathcal{M}$, $s \in S$ and its corresponding state $\tilde{s} \in \tilde{S}$, there exists a strategy $\tilde{\sigma}$ in $\mathcal{M}_\lambda$ such that the probability of satisfying the direct fixed (resp. fixed) window objective of maximal window size λ in $\mathcal{M}_s^\sigma$ equals the probability of satisfying the safety (resp. co-Büchi) objective in $\mathcal{M}_{\lambda,\tilde{s}}^{\tilde{\sigma}}$, and conversely. To obtain a strategy σ in $\mathcal{M}$ from a strategy $\tilde{\sigma}$ in $\mathcal{M}_\lambda$, we have to integrate in the memory of σ the additional information encoded in $\tilde{S}$: hence the memory required by σ is the one used by $\tilde{\sigma}$ with a blow-up polynomial in $|\tilde{S}|$. These reductions, along with the fact that pure memoryless strategies suffice for safety and co-Büchi objectives in MDPs [4], yield sufficiency of finite-memory strategies, a *key ingredient* in our generic approach (Lem. 10).

▶ **Theorem 4.** *Pure finite-memory strategies suffice for the threshold probability problem for all fixed window objectives. That is, given MDP $\mathcal{M} = (S, A, \delta)$, initial state $s \in S$, window size $\lambda > 0$, $\Omega \in \{\mathsf{mp}, \mathsf{par}\}$, objective $E \in \{\mathsf{DFW}_\Omega(\lambda), \mathsf{FW}_\Omega(\lambda)\}$ and threshold probability $\alpha \in [0,1] \cap \mathbb{Q}$, if there exists a strategy $\sigma \in \Sigma$ such that $\mathbb{P}^\sigma_{\mathcal{M},s}[E] \geq \alpha$, then there exists a pure finite-memory strategy σ' such that $\mathbb{P}^{\sigma'}_{\mathcal{M},s}[E] \geq \alpha$.*

These reductions also yield *algorithms* for the fixed window case. We only use them for the direct variants, as the approach we develop in Sect. 6 proves to be more efficient for the prefix-independent one, for two reasons: first, we may restrict the co-Büchi-like analysis to ECs; second, we use a more tractable analysis than the co-Büchi unfolding for mean-payoff.

We complement the corresponding upper bounds with almost-matching lower bounds, showing that our approach is close to optimal, complexity-wise.

▶ **Theorem 5.** *The threshold probability problem is*

(a) P*-complete for direct fixed window parity objectives, and pure polynomial-memory optimal strategies can be constructed in polynomial time. Furthermore, polynomial memory is in general necessary.*

(b) *in* $\mathsf{EXPTIME}$ *and* PSPACE*-hard for direct fixed window mean-payoff objectives (already for acyclic MDPs), and pure pseudo-polynomial-memory optimal strategies can be constructed in pseudo-polynomial time. Furthermore, pseudo-polynomial memory is in general necessary.*

Upper bounds. The algorithm is simple: given $\mathcal{M}$ and $\lambda > 0$, build $\mathcal{M}_\lambda$ and solve the corresponding safety problem. This can be done in polynomial time in $|\mathcal{M}_\lambda|$ and pure memoryless strategies suffice over $\mathcal{M}_\lambda$ [4]. For parity, $\mathcal{M}_\lambda$ is of size polynomial in $|\mathcal{M}|$, d and λ. Since both d (anyway bounded by $\mathcal{O}(|S|)$) and λ are assumed to be given in unary, it yields the result. For mean-payoff, $\mathcal{M}_\lambda$ is of size polynomial in $|\mathcal{M}|$, W and λ. Since weights are assumed to be encoded in binary, we only have a pseudo-polynomial-time algorithm.

Lower bounds. We give some insights about the reductions yielding the results for lower complexity bounds. We begin with **P-hardness** (item (a)). Roughly, we reduce two-player reachability games [6, 33] to direct fixed window parity MDPs, using two key ingredients: (i) if winning is possible in the game, it is possible in bounded time: we deduce a sufficient window size λ from it; (ii) *almost-sure* winning for $\mathsf{DFW}_\Omega(\lambda)$ objectives is equivalent to *sure* winning (if a losing run exists, it is witnessed by a finite prefix of strictly positive probability).

Regarding memory, the proof established for direct fixed window parity games in [14] carries over easily to our setting by replacing the states of the opponent by stochastic actions, in the natural way. Hence the lower bound is trivial to establish.

Consider now **PSPACE-hardness** (item (b)). We proceed via a reduction from the threshold probability problem for shortest path objectives [30, 37]. Given an MDP $\mathcal{M}$ with state space S, a lower probability bound α and an upper bound $\ell \in \mathbb{N}$ on the cumulative sum of weights of actions through runs of the system, this problem asks whether there exists a strategy allowing to visit a target set $T \subseteq S$ with probability at least α and cost at most ℓ (note that weights are assumed to be strictly positive in this setting). The problem is known to be PSPACE-hard, even for *acyclic* MDPs [30].

We establish a reduction from this problem, in the acyclic case, to a threshold probability problem for $\mathsf{DFW}_{\mathsf{mp}}(\lambda = |S|)$, maintaining the acyclicity of the underlying graph. From $\mathcal{M}$, we build a new MDP $\mathcal{M}'$ by taking the opposite of all weights; adding the bound ℓ when entering the target; and making the target cost-free. The result follows from three key ingredients: (i) the sum of weights over a prefix in $\mathcal{M}'$ that is not yet in T is strictly negative, and the opposite of the sum over the same prefix in $\mathcal{M}$; (ii) due to the addition of ℓ on entering T, any run of $\mathcal{M}'$ sees all its windows closed if and only if T is reached with a cost less than ℓ in $\mathcal{M}$; (iii) using the acyclicity, if a run reaches T, it does so within λ steps.

The need for pseudo-polynomial memory is also proved through this reduction. Indeed, there is a chain of reductions from *subset-sum games* [39, 24] to our setting, via the shortest path problem [30]. Subset-sum games require pseudo-polynomial-memory strategies.

▶ Remark 6. As noted above, almost-surely winning coincides with surely winning for the *direct fixed* window objectives. Therefore, the threshold probability problem for $\mathsf{DFW}_{\mathsf{mp}}(\lambda)$ collapses to P if $\alpha = 1$ [17].

5 The case of end-components

We have solved the case of direct fixed window objectives: it remains to consider prefix-independent fixed and bounded variants. The analysis of MDPs with prefix-independent objectives crucially relies on ECs (Sect. 2): they are almost-surely reached in the long run.

First, we study what happens in ECs: how to play optimally and what can be achieved. In Sect. 6, we will use this knowledge as the cornerstone of our algorithm for general MDPs. The main result here is a *strong link between ECs and two-player games*: intuitively, either the probability to win a window objective in an EC is zero, or it is one and there exists a sub-EC where the controller can actually win surely, i.e., as in a two-player game played on this sub-EC. We start by defining the notion of λ-safety, that will characterize such sub-ECs.

▶ **Definition 7** (λ-safety). *Let $\mathcal{M}$ be an MDP, $\Omega \in \{\mathsf{mp}, \mathsf{par}\}$, $\lambda > 0$, and $\mathcal{C} = (S_{\mathcal{C}}, A_{\mathcal{C}}, \delta_{\mathcal{C}}) \in \mathsf{EC}(\mathcal{M})$, we say that $\mathcal{C}$ is λ-safe$_\Omega$ if there exists a strategy $\sigma \in \Sigma$ in $\mathcal{C}$ such that, from all $s \in S_{\mathcal{C}}$, $\mathsf{S}^{\sigma}_{\mathcal{C},s}[\mathsf{DFW}_\Omega(\lambda)]$.*

Classifying an EC as λ-safe$_\Omega$ or not boils down to interpreting it as a *two-player game* (the duality between MDPs and games is further explored in [13, 7]). The uncertainty becomes adversarial: on entering a state s of the MDP, the controller chooses an action a following its strategy and the opponent then chooses a successor s' such that $s' \in \mathsf{Supp}(\delta(s,a))$ without taking into account the exact values of probabilities. In such a view, the opponent tries to prevent the controller from achieving its objective. A *winning strategy* for the controller in the game interpretation is a strategy that ensures the objective regardless of its opponent's strategy. An EC is thus said to be λ-safe$_\Omega$ if and only if its two-player interpretation admits a winning strategy for $\mathsf{DFW}_\Omega(\lambda)$. W.l.o.g., all our strategies are uniform in the game-theoretic sense: we use it in our statements.

▶ **Proposition 8.** *Let $\mathcal{M}$ be an MDP, $\Omega \in \{\mathsf{mp}, \mathsf{par}\}$, $\lambda > 0$, and $\mathcal{C} = (S_{\mathcal{C}}, A_{\mathcal{C}}, \delta_{\mathcal{C}}) \in \mathsf{EC}(\mathcal{M})$ be λ-safe$_\Omega$. Then, there exists a pure polynomial-memory strategy $\sigma_{safe}^{\Omega,\lambda,\mathcal{C}}$ in $\mathcal{C}$ such that $\mathsf{S}_{\mathcal{C},s}^{\sigma_{safe}^{\Omega,\lambda,\mathcal{C}}}[\mathsf{DFW}_\Omega(\lambda)]$ for all $s \in S_{\mathcal{C}}$.*

The proof is straightforward by definition of λ-safety and pure polynomial-memory strategies being sufficient in direct fixed window games, both for mean-payoff [17] and parity [14].

As sketched above, the existence of sub-ECs that are λ-safe is crucial in order to satisfy any window objective in an EC. We thus introduce the notion of *good* ECs.

▶ **Definition 9.** *Let $\mathcal{M}$ be an MDP, $\Omega \in \{\mathsf{mp}, \mathsf{par}\}$, and $\mathcal{C} \in \mathsf{EC}(\mathcal{M})$, we say that*

- *$\mathcal{C}$ is λ-good$_\Omega$, for $\lambda > 0$, if it contains a sub-EC $\mathcal{C}'$ which is λ-safe$_\Omega$.*
- *$\mathcal{C}$ is BW-good$_\Omega$ if it contains a sub-EC $\mathcal{C}'$ which is λ-safe$_\Omega$ for some $\lambda > 0$.*

Any BW-good$_\Omega$ EC is λ-good$_\Omega$ for an appropriate $\lambda > 0$. Yet, we use a different terminology as in the BW case, we do not fix λ a priori: this is important complexity-wise.

We now establish that good$_\Omega$ ECs are exactly the ones where window objectives can be satisfied with non-zero probability, and actually, with probability one.

▶ **Lemma 10** (Zero-one law)**.** *Let $\mathcal{M}$ be an MDP, $\Omega \in \{\mathsf{mp}, \mathsf{par}\}$ and $\mathcal{C} = (S_{\mathcal{C}}, A_{\mathcal{C}}, \delta_{\mathcal{C}}) \in \mathsf{EC}(\mathcal{M})$. The following assertions hold.*

(a) *For all $\lambda > 0$,*
  - **(i)** *either $\mathcal{C}$ is λ-good$_\Omega$ and there exists a strategy σ in $\mathcal{C}$ such that $\mathsf{AS}_{\mathcal{C},s}^{\sigma}[\mathsf{FW}_\Omega(\lambda)]$ for all $s \in S_{\mathcal{C}}$,*
  - **(ii)** *or for all $s \in S_{\mathcal{C}}$ and for all strategy σ in $\mathcal{C}$, $\mathbb{P}_{\mathcal{C},s}^{\sigma}[\mathsf{FW}_\Omega(\lambda)] = 0$.*

(b)
  - **(i)** *Either $\mathcal{C}$ is BW-good$_\Omega$ and there exists a strategy σ in $\mathcal{C}$ such that $\mathsf{AS}_{\mathcal{C},s}^{\sigma}[\mathsf{BW}_\Omega]$ for all $s \in S_{\mathcal{C}}$, or*
  - **(ii)** *for all $s \in S_{\mathcal{C}}$, for all strategy σ in $\mathcal{C}$, $\mathbb{P}_{\mathcal{C},s}^{\sigma}[\mathsf{BW}_\Omega] = 0$.*

We sketch the proof by focusing on case (a). Roughly, (i) holds thanks to the two following facts. First, $\mathcal{C}$ is an EC in which there exists a strategy almost-surely visiting all states of its state space. Second, there is a λ-safe$_\Omega$ sub-EC in $\mathcal{C}$ in which there exists a strategy surely satisfying $\mathsf{DFW}_\Omega(\lambda)$. Combining these two strategies yields the result. For case (ii), recall that finite-memory strategies suffice for $\mathsf{FW}_\Omega(\lambda)$ objectives by Thm. 4. Hence, we fix a finite-memory strategy in $\mathcal{C}$, yielding a finite induced MC where runs almost-surely end up in a BSCC $\mathcal{B}$ [4]. There is no λ-safe$_\Omega$ sub-EC, so there exists a run $\widehat{\rho}$ in $\mathcal{B}$ such that $\widehat{\rho} \notin \mathsf{DFW}_\Omega(\lambda)$. From $\widehat{\rho}$, we extract a history $\widehat{h}$ that contains a window open for λ steps. Since all states in $\mathcal{B}$ are almost-surely visited infinitely often, $\hat{h}$ also happens infinitely often with probability one and the probability to win $\mathsf{FW}_\Omega(\lambda)$ when reaching $\mathcal{B}$ is thus zero. Since this holds for any BSCC induced by σ, we obtain the claim.

Items (b)(i) and (b)(ii) can then be shown by using the fact that $\mathsf{BW}_\Omega = \bigcup_{\lambda>0} \mathsf{FW}_\Omega(\lambda)$.

▶ Remark 11. An interesting consequence of Lem. 10 is the existence of uniform bounds on λ in ECs, in contrast to the general MDP case, as seen in Sect. 3. This is indeed natural, as we established that winning with positive probability within an EC coincides with winning surely in a sub-EC; sub-EC that can be seen as a two-player zero-sum game where uniform bounds are granted by [17, 14].

Lem. 10 establishes that interesting strategies exist in good$_\Omega$ ECs. Let us describe them.

▶ **Proposition 12.** *Let $\mathcal{M}$ be an MDP, $\Omega \in \{\mathsf{mp}, \mathsf{par}\}$, and $\mathcal{C} = (S_{\mathcal{C}}, A_{\mathcal{C}}, \delta_{\mathcal{C}}) \in \mathsf{EC}(\mathcal{M})$.*

- *If $\mathcal{C}$ is λ-good$_\Omega$, for $\lambda > 0$, then there exists a pure polynomial-memory strategy $\sigma_{good}^{\Omega,\lambda,\mathcal{C}}$ such that $\mathsf{AS}_{\mathcal{C},s}^{\sigma_{good}^{\Omega,\lambda,\mathcal{C}}}[\mathsf{FW}_\Omega(\lambda)]$ for all $s \in S_{\mathcal{C}}$.*

- *If $\mathcal{C}$ is* BW*-good*$_\Omega$*, then there exists a pure memoryless strategy* $\sigma_{good}^{\Omega,\mathsf{BW},\mathcal{C}}$ *such that* $\mathsf{AS}_{\mathcal{C},s}^{\sigma_{good}^{\Omega,\mathsf{BW},\mathcal{C}}}[\mathsf{BW}_\Omega]$ *for all* $s \in S_{\mathcal{C}}$.

Intuitively, such strategies first mimic a pure memoryless strategy reaching a safe$_\Omega$ sub-EC almost-surely, then switch to a strategy surely winning in this sub-EC, which is lifted from the game interpretation.

We may already sketch a general solution to the threshold probability problem based on Lem. 10 and the well-known fact that ECs are almost-surely reached under any strategy: an optimal strategy must maximize the probability to reach good$_\Omega$ ECs. It is therefore crucial to be able to identify such ECs efficiently. However, an MDP may contain an exponential number of ECs. Fortunately, the next lemma states that we do not have to test them all.

▶ **Lemma 13.** *Let $\mathcal{M}$ be an MDP and $\mathcal{C} \in \mathsf{EC}(\mathcal{M})$. If $\mathcal{C}$ is λ-good$_\Omega$ (resp.* BW*-good*$_\Omega$*), then it is also the case of any super-EC $\mathcal{C}' \in \mathsf{EC}(\mathcal{M})$ containing $\mathcal{C}$.*

▶ **Corollary 14.** *Let $\mathcal{M}$ be an MDP and $\mathcal{C} \in \mathsf{MEC}(\mathcal{M})$ be a maximal EC. If $\mathcal{C}$ is not λ-good$_\Omega$ (resp.* BW*-good*$_\Omega$*), then neither is any of its sub-EC $\mathcal{C}' \in \mathsf{EC}(\mathcal{M})$.*

The interest of Cor. 14 is that the number of MECs is bounded by $|S|$ for any MDP $\mathcal{M} = (S, A, \delta)$ because they are all disjoint. Furthermore, decomposing $\mathcal{M}$ in MECs can be done efficiently (e.g., quadratic time [18]). So, we know classifying MECs is sufficient and MECs can easily be identified: it remains to discuss how to classify a MEC as good$_\Omega$ or not.

Let $\mathcal{M} = (S, A, \delta)$. Recall that a MEC $\mathcal{C} = (S_{\mathcal{C}}, A_{\mathcal{C}}, \delta_{\mathcal{C}}) \in \mathsf{MEC}(\mathcal{M})$ is λ-good$_\Omega$ (resp. BW-good$_\Omega$) if and only if it contains a λ-safe$_\Omega$ sub-EC. This is equivalent to having a non-empty *winning set* for the controller in the two-player game over $\mathcal{C}$ – naturally defined as above. This winning set contains all states in $S_{\mathcal{C}}$ from which the controller has a *surely winning* strategy. This set, if non-empty, contains at least one sub-EC of $\mathcal{C}$, as otherwise the opponent could force the controller to leave it and win the game (by prefix-independence). Thus, testing if a MEC is good$_\Omega$ boils down to solving its two-player game interpretation [17, 14].

▶ **Theorem 15** (MEC classification)**.** *Let $\mathcal{M}$ be an MDP and $\mathcal{C} \in \mathsf{MEC}(\mathcal{M})$. The following assertions hold.*

(a) *Deciding if $\mathcal{C}$ is λ-good$_\Omega$, for $\lambda > 0$, is in* P *for $\Omega \in \{\mathsf{mp}, \mathsf{par}\}$ and a corresponding pure polynomial-memory strategy* $\sigma_{good}^{\Omega,\lambda,\mathcal{C}}$ *can be constructed in polynomial time.*

(b) *Deciding if $\mathcal{C}$ is* BW*-good*$_{\mathsf{mp}}$ *is in* $\mathsf{NP} \cap \mathsf{coNP}$ *and a corresponding pure memoryless strategy* $\sigma_{good}^{\mathsf{mp},\mathsf{BW},\mathcal{C}}$ *can be constructed in pseudo-polynomial time.*

(c) *Deciding if $\mathcal{C}$ is* BW*-good*$_{\mathsf{par}}$ *is in* P *and a corresponding pure memoryless strategy* $\sigma_{good}^{\mathsf{par},\mathsf{BW},\mathcal{C}}$ *can be constructed in polynomial time.*

6 General MDPs

We have all the ingredients to establish an algorithm in the general case. Given an MDP $\mathcal{M}$, a state s and a window objective $\mathsf{FW}_\Omega(\lambda)$ for $\lambda > 0$ (resp. BW_Ω), we (i) compute the MEC decomposition of $\mathcal{M}$; (ii) classify each MEC as λ-good$_\Omega$ (resp. BW-good$_\Omega$) or not; and (iii) compute an optimal strategy from s to reach the union of good$_\Omega$ MECs: the probability of reaching such MECs is exactly the maximum probability for the window objective.

The fixed and bounded versions are presented in Fig. 2. Sub-procedure $\mathsf{MaxReachability}(s, T)$ computes the maximum probability to reach the set T from s (in polynomial time [4]). The overall complexity of the algorithm is dominated by the classification step.

Algorithm 1 FixedWindow$(\mathcal{M}, s, \Omega, \lambda)$.

Input: MDP $\mathcal{M}$, state s, $\Omega \in \{\mathsf{mp}, \mathsf{par}\}$, $\lambda > 0$
Output: Maximum probability of $\mathsf{FW}_\Omega(\lambda)$ from s

$T \leftarrow \emptyset$
for all $\mathcal{C} = (S_\mathcal{C}, A_\mathcal{C}, \delta_\mathcal{C}) \in \mathsf{MEC}(\mathcal{M})$ **do**
  if $\mathcal{C}$ is λ-good$_\Omega$ **then**
    $T \leftarrow T \uplus S_\mathcal{C}$
$\nu = \mathsf{MaxReachability}(s, T)$
return ν

Algorithm 2 BoundedWindow$(\mathcal{M}, s, \Omega)$.

Input: MDP $\mathcal{M}$, state s, $\Omega \in \{\mathsf{mp}, \mathsf{par}\}$
Output: Maximum probability of BW_Ω from s

$T \leftarrow \emptyset$
for all $\mathcal{C} = (S_\mathcal{C}, A_\mathcal{C}, \delta_\mathcal{C}) \in \mathsf{MEC}(\mathcal{M})$ **do**
  if $\mathcal{C}$ is BW-good$_\Omega$ **then**
    $T \leftarrow T \uplus S_\mathcal{C}$
$\nu = \mathsf{MaxReachability}(s, T)$
return ν

Figure 2 Algorithms computing the max. probability of prefix independent window objectives.

▶ **Lemma 16.** *Alg. 1 and Alg. 2 are correct: given an MDP $\mathcal{M} = (S, A, \delta)$, an initial state $s \in S$, $\Omega \in \{\mathsf{mp}, \mathsf{par}\}$, $\lambda > 0$, we have that* $\mathsf{FixedWindow}(\mathcal{M}, s, \Omega, \lambda) = \max_{\sigma \in \Sigma} \mathbb{P}^\sigma_{\mathcal{M},s}[\mathsf{FW}_\Omega(\lambda)]$ *and* $\mathsf{BoundedWindow}(\mathcal{M}, s, \Omega) = \max_{\sigma \in \Sigma} \mathbb{P}^\sigma_{\mathcal{M},s}[\mathsf{BW}_\Omega]$.

The proof is straightforward based on our previous results: (i) using prefix-independence and almost-sure reachability of MECs, we know that the highest probability (of satisfying the objective) is obtained when the "best" MECs are reached; (ii) by Lem. 10, this probability is one in good$_\Omega$ ECs and zero in the others; (iii) maximizing the probability to reach good$_\Omega$ ECs is exactly how our algorithms operate.

▶ **Theorem 17.** *The threshold probability problem is*

(a) P*-complete for fixed window parity and fixed window mean-payoff objectives, and pure polynomial-memory optimal strategies can be constructed in polynomial time. Furthermore, polynomial memory is in general necessary.*

(b) P*-complete for bounded window parity objectives, and pure memoryless optimal strategies can be constructed in polynomial time;*

(c) *in* NP $\cap$ coNP *and as hard as mean-payoff games for bounded window mean-payoff objectives, and pure memoryless optimal strategies can be constructed in pseudo-polynomial time.*

The results follow from Thm. 15 and the MEC classification in the $\mathsf{BW}\text{-good}_{\mathsf{mp}}$ case being the only non-polynomial operation of our algorithms. Plugging pure memoryless optimal strategies for reaching good$_\Omega$ MECs (granted by [4]) to our MEC strategies yields the upper bounds on memory. Hardness is essentially obtained through the two-player game interpretation of ECs [14, 17].

7 Limitations and perspectives

We summarized our results and compared them to the state of the art in Sect. 1. Here, we discuss the limitations of our work and some extensions within arm's reach.

Direct bounded window objectives. We left out a window objective considered in games [17, 14]: the *direct bounded* one, $\mathsf{DBW}_\Omega = \bigcup_{\lambda>0} \mathsf{DFW}_\Omega(\lambda)$. It is maybe not the most natural as it is *not* prefix-independent, yet allows to close the windows of a run in an arbitrarily large number of steps bounded along the run. This variant gives rise to complex behaviors in MDPs, notably due to its interaction with the almost-sure reachability of ECs.

Consider the MDP in Fig. 3 and objective $\mathsf{DBW}_{\mathsf{mp}}$. A window opens due to a. The only way to close it is to use b up to the point where the running sum becomes non-negative. When it does, all windows are closed and the controller may switch to s_5. Observe that

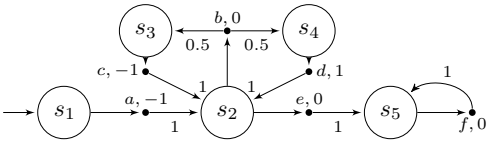

Figure 3 There exists a strategy σ ensuring $\mathsf{AS}^{\sigma}_{\mathcal{M},s_1}[\mathsf{DBW}_{\mathsf{mp}}]$ but it requires infinite memory as it needs to use b up to the point where the running sum becomes non-negative, then switch to e.

taking b repeatedly induces a *symmetric random walk* [29]. Classical probability results ensure that a non-negative sum will be obtained almost-surely, but the number of times b is played must remain unbounded (as for any bounded number, there exists a strictly positive probability to obtain only -1's for example). Thus, there exists an infinite-memory strategy σ such that $\mathsf{AS}^{\sigma}_{\mathcal{M},s_1}[\mathsf{DBW}_{\mathsf{mp}}]$, but no finite-memory one can do as good.

Now, let $\delta(s_2, b) = \{s_3 \mapsto 0.6, s_4 \mapsto 0.4\}$: the random walk becomes *asymmetric*, with a strictly positive chance to diverge toward $-\infty$. While the best possible strategy is still the one defined above, it only satisfies the objective with probability strictly less than one.

What do we observe? First, *infinite-memory strategies are required*, which is a problem for practical applications. Second, even for *qualitative* questions (is the probability zero or one?), the actual probabilities of the MDP must be considered, not only the existence of a transition. This is in stark contrast to most problems in MDPs [4]: in that sense, the direct bounded window objective is not well-behaved. This is due to the above connection with random walks. It is well-known that complex random walks are difficult to tackle for verification and synthesis: e.g., even simple asymmetric random walks are not *decisive* MCs, a large and robust class of MCs where reachability questions can be answered [1].

Markov chains. Our work focuses on the threshold probability problem for MDPs, and the corresponding strategy synthesis problem. Better complexities could possibly be obtained for MCs, where there are no non-deterministic choices. To achieve this, a natural direction would be to focus on the classification of ECs (Sect. 5), the bottleneck of our approach: for MCs, this classification would involve *one-player window games* (for the opponent), whose complexity has yet to be explored and would certainly be lower than for two-player games.

However, complexity is unlikely to be much lower: all parity variants are already in P, and the high complexity of $\mathsf{DFW}_{\mathsf{mp}}$ would remain: a construction similar to the PSPACE-hardness (Thm. 5) easily shows this problem to be PP-hard, already for *acyclic* MCs (again using [30]).

Expected value problem. Given an MDP $\mathcal{M}$ and an initial state s, we may be interested in synthesizing a strategy σ that minimizes the *expected window size* for a fixed window objective (say $\mathsf{FW}_\Omega(\lambda)$), which we straightforwardly define as $\mathbb{E}^{\sigma}_{\mathcal{M},s,\Omega}(\lambda) = \sum_{\lambda>0}^{\infty} \lambda \cdot \mathbb{P}^{\sigma}_{\mathcal{M},s}[\mathsf{FW}_\Omega(\lambda) \setminus \mathsf{FW}_\Omega(\lambda-1)]$, with $\mathsf{FW}_\Omega(0) = \emptyset$. This meets the natural desire to build strategies that strive to maintain the *best time bounds possible in their local environment* (e.g., EC of $\mathcal{M}$). Note that this is totally different from the value function used in [8].

For prefix-independent variants, *we already have all the necessary machinery* to solve this problem. First, we refine the classification process to identify the best window size achievable in each MEC, if any. Indeed, if a MEC is λ-good, it necessarily is for some λ between one and the upper bound derived from the game-theoretic interpretation (Rmk. 11): we determine the smallest value of λ for each MEC via a binary search coupled with the classification procedure. Second, using classical techniques (e.g., [37]), we contract each MEC to a single-state EC, and give it a weight that represents the best window size we can ensure in it (hence this

weight may be infinite if a MEC is not BW-good). Finally, we construct a global strategy that favors reaching MECs with the lowest weights, for example by synthesizing a strategy minimizing the classical mean-payoff value. Note that if λ-good MECs cannot be reached almost-surely, the expected value will be infinite, as wanted. Observe that *such an approach maintains tractability*, as we end up with a polynomial-time algorithm.

Direct variants require more involved techniques, as the unfoldings of Sect. 4 are strongly linked to the window size λ, and cannot be easily combined for different values of λ.

Multi-objective problems. Window games have been studied in the multidimension setting, where several weight (resp. priority) functions are given, and the objective is the intersection of all one-dimension objectives [17, 14]. Again, *our generic approach supports effortless extension to this setting.* In the direct case, unfoldings of Sect. 4 can be generalized to multiple dimensions, as in [17, 14]. For prefix-independent variants, the EC classification needs to be adapted to handle multidimension window games, which we can solve using [17, 14]. Then, we also need to consider a multi-objective reachability problem [37]. While almost all cases of multidimension window games are EXPTIME-complete, decidability of the bounded mean-payoff case is still open, but however known to be non-primitive recursive hard.

Tool support. Thanks to its low complexity and its adequacy w.r.t. applications, our window framework lends itself well to tool development. We are currently building a tool suite for MDPs with window objectives based on the main results of this paper along with the aforementioned extensions. Our aim is to provide a dedicated extension of Storm, a cutting-edge probabilistic model checker [23].

References

1 Parosh Aziz Abdulla, Noomene Ben Henda, and Richard Mayr. Decisive Markov Chains. *Logical Methods in Computer Science*, 3(4), 2007. doi:10.2168/LMCS-3(4:7)2007.

2 Pranav Ashok, Krishnendu Chatterjee, Przemyslaw Daca, Jan Kretínský, and Tobias Meggendorfer. Value Iteration for Long-Run Average Reward in Markov Decision Processes. In Rupak Majumdar and Viktor Kuncak, editors, *Computer Aided Verification - 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part I*, volume 10426 of *Lecture Notes in Computer Science*, pages 201–221. Springer, 2017. doi:10.1007/978-3-319-63387-9_10.

3 Christel Baier. Reasoning About Cost-Utility Constraints in Probabilistic Models. In Mikolaj Bojanczyk, Slawomir Lasota, and Igor Potapov, editors, *Reachability Problems - 9th International Workshop, RP 2015, Warsaw, Poland, September 21-23, 2015, Proceedings*, volume 9328 of *Lecture Notes in Computer Science*, pages 1–6. Springer, 2015. doi:10.1007/978-3-319-24537-9_1.

4 Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT press, 2008.

5 Christel Baier, Joachim Klein, Sascha Klüppelholz, and Sascha Wunderlich. Weight monitoring with linear temporal logic: complexity and decidability. In Thomas A. Henzinger and Dale Miller, editors, *Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), CSL-LICS '14, Vienna, Austria, July 14 - 18, 2014*, pages 11:1–11:10. ACM, 2014. doi:10.1145/2603088.2603162.

6 Catriel Beeri. On the Membership Problem for Functional and Multivalued Dependencies in Relational Databases. *ACM Trans. Database Syst.*, 5(3):241–259, 1980. doi:10.1145/320613.320614.

7 Raphaël Berthon, Mickael Randour, and Jean-François Raskin. Threshold Constraints with Guarantees for Parity Objectives in Markov Decision Processes. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80 of *LIPIcs*, pages 121:1–121:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017. doi:10.4230/LIPIcs.ICALP.2017.121.

8 Benjamin Bordais, Shibashis Guha, and Jean-François Raskin. Expected Window Mean-Payoff. *CoRR*, abs/1812.09298, 2018. arXiv:1812.09298.

9 Patricia Bouyer, Mauricio González, Nicolas Markey, and Mickael Randour. Multi-weighted Markov Decision Processes with Reachability Objectives. In Andrea Orlandini and Martin Zimmermann, editors, *Proceedings Ninth International Symposium on Games, Automata, Logics, and Formal Verification, GandALF 2018, Saarbrücken, Germany, 26-28th September 2018.*, volume 277 of *EPTCS*, pages 250–264, 2018. doi:10.4204/EPTCS.277.18.

10 Tomás Brázdil, Krishnendu Chatterjee, Vojtech Forejt, and Antonín Kucera. Trading performance for stability in Markov decision processes. *J. Comput. Syst. Sci.*, 84:144–170, 2017. doi:10.1016/j.jcss.2016.09.009.

11 Tomás Brázdil, Vojtech Forejt, Antonín Kucera, and Petr Novotný. Stability in Graphs and Games. In Josée Desharnais and Radha Jagadeesan, editors, *27th International Conference on Concurrency Theory, CONCUR 2016, August 23-26, 2016, Québec City, Canada*, volume 59 of *LIPIcs*, pages 10:1–10:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016. doi:10.4230/LIPIcs.CONCUR.2016.10.

12 Thomas Brihaye, Florent Delgrange, Youssouf Oualhadj, and Mickael Randour. Life is Random, Time is Not: Markov Decision Processes with Window Objectives. *CoRR*, abs/1901.03571, 2019. arXiv:1901.03571.

13 Véronique Bruyère, Emmanuel Filiot, Mickael Randour, and Jean-François Raskin. Meet your expectations with guarantees: Beyond worst-case synthesis in quantitative games. *Inf. Comput.*, 254:259–295, 2017. doi:10.1016/j.ic.2016.10.011.

14 Véronique Bruyère, Quentin Hautem, and Mickael Randour. Window parity games: an alternative approach toward parity games with time bounds. In Domenico Cantone and Giorgio Delzanno, editors, *Proceedings of the Seventh International Symposium on Games, Automata, Logics and Formal Verification, GandALF 2016, Catania, Italy, 14-16 September 2016.*, volume 226 of *EPTCS*, pages 135–148, 2016. doi:10.4204/EPTCS.226.10.

15 Véronique Bruyère, Quentin Hautem, and Jean-François Raskin. On the Complexity of Heterogeneous Multidimensional Games. In Josée Desharnais and Radha Jagadeesan, editors, *27th International Conference on Concurrency Theory, CONCUR 2016, August 23-26, 2016, Québec City, Canada*, volume 59 of *LIPIcs*, pages 11:1–11:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016. doi:10.4230/LIPIcs.CONCUR.2016.11.

16 Cristian S. Calude, Sanjay Jain, Bakhadyr Khoussainov, Wei Li, and Frank Stephan. Deciding parity games in quasipolynomial time. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 252–263. ACM, 2017. doi:10.1145/3055399.3055409.

17 Krishnendu Chatterjee, Laurent Doyen, Mickael Randour, and Jean-François Raskin. Looking at mean-payoff and total-payoff through windows. *Inf. Comput.*, 242:25–52, 2015. doi:10.1016/j.ic.2015.03.010.

18 Krishnendu Chatterjee and Monika Henzinger. Efficient and Dynamic Algorithms for Alternating Büchi Games and Maximal End-Component Decomposition. *J. ACM*, 61(3):15:1–15:40, 2014. doi:10.1145/2597631.

19 Krishnendu Chatterjee, Thomas A. Henzinger, and Florian Horn. Finitary winning in omega-regular games. *ACM Trans. Comput. Log.*, 11(1):1:1–1:27, 2009. doi:10.1145/1614431.1614432.

20 Krishnendu Chatterjee, Marcin Jurdzinski, and Thomas A. Henzinger. Quantitative stochastic parity games. In J. Ian Munro, editor, *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004, New Orleans, Louisiana, USA, January 11-14, 2004*, pages 121–130. SIAM, 2004. URL: `http://dl.acm.org/citation.cfm?id=982792.982808`.

21 Wojciech Czerwinski, Laure Daviaud, Nathanaël Fijalkow, Marcin Jurdzinski, Ranko Lazic, and Pawel Parys. Universal trees grow inside separating automata: Quasi-polynomial lower bounds for parity games. *CoRR*, abs/1807.10546, 2018. `arXiv:1807.10546`.

22 Laure Daviaud, Marcin Jurdzinski, and Ranko Lazic. A pseudo-quasi-polynomial algorithm for mean-payoff parity games. In Anuj Dawar and Erich Grädel, editors, *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2018, Oxford, UK, July 09-12, 2018*, pages 325–334. ACM, 2018. `doi:10.1145/3209108.3209162`.

23 Christian Dehnert, Sebastian Junges, Joost-Pieter Katoen, and Matthias Volk. A Storm is Coming: A Modern Probabilistic Model Checker. In Rupak Majumdar and Viktor Kuncak, editors, *Computer Aided Verification - 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part II*, volume 10427 of *Lecture Notes in Computer Science*, pages 592–600. Springer, 2017. `doi:10.1007/978-3-319-63390-9_31`.

24 John Fearnley and Marcin Jurdzinski. Reachability in two-clock timed automata is PSPACE-complete. *Inf. Comput.*, 243:26–36, 2015. `doi:10.1016/j.ic.2014.12.004`.

25 Nathanaël Fijalkow, Pawel Gawrychowski, and Pierre Ohlmann. The complexity of mean payoff games using universal graphs. *CoRR*, abs/1812.07072, 2018. `arXiv:1812.07072`.

26 Jerzy Filar and Koos Vrieze. *Competitive Markov decision processes*. Springer, 1997.

27 Thomas Gawlitza and Helmut Seidl. Games through Nested Fixpoints. In Ahmed Bouajjani and Oded Maler, editors, *Computer Aided Verification, 21st International Conference, CAV 2009, Grenoble, France, June 26 - July 2, 2009. Proceedings*, volume 5643 of *Lecture Notes in Computer Science*, pages 291–305. Springer, 2009. `doi:10.1007/978-3-642-02658-4_24`.

28 Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata, Logics, and Infinite Games: A Guide to Current Research [outcome of a Dagstuhl seminar, February 2001]*, volume 2500 of *Lecture Notes in Computer Science*. Springer, 2002. `doi:10.1007/3-540-36387-4`.

29 Charles M. Grinstead and J. Laurie Snell. *Introduction to probability*. American Mathematical Society, 1997.

30 Christoph Haase and Stefan Kiefer. The Odds of Staying on Budget. In Magnús M. Halldórsson, Kazuo Iwama, Naoki Kobayashi, and Bettina Speckmann, editors, *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part II*, volume 9135 of *Lecture Notes in Computer Science*, pages 234–246. Springer, 2015. `doi:10.1007/978-3-662-47666-6_19`.

31 Arnd Hartmanns, Sebastian Junges, Joost-Pieter Katoen, and Tim Quatmann. Multi-cost Bounded Reachability in MDP. In Dirk Beyer and Marieke Huisman, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 24th International Conference, TACAS 2018, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2018, Thessaloniki, Greece, April 14-20, 2018, Proceedings, Part II*, volume 10806 of *Lecture Notes in Computer Science*, pages 320–339. Springer, 2018. `doi:10.1007/978-3-319-89963-3_19`.

32 Paul Hunter, Guillermo A. Pérez, and Jean-François Raskin. Looking at mean payoff through foggy windows. *Acta Inf.*, 55(8):627–647, 2018. `doi:10.1007/s00236-017-0304-7`.

33 Neil Immerman. Number of Quantifiers is Better Than Number of Tape Cells. *J. Comput. Syst. Sci.*, 22(3):384–406, 1981. `doi:10.1016/0022-0000(81)90039-8`.

34 Marcin Jurdzinski. Deciding the Winner in Parity Games is in UP ∩ co-UP. *Inf. Process. Lett.*, 68(3):119–124, 1998. `doi:10.1016/S0020-0190(98)00150-1`.

35 Mickael Randour. Automated Synthesis of Reliable and Efficient Systems Through Game Theory: A Case Study. In *Proc. of ECCS 2012*, Springer Proceedings in Complexity XVII, pages 731–738. Springer, 2013. `doi:10.1007/978-3-319-00395-5_90`.

36 Mickael Randour, Jean-François Raskin, and Ocan Sankur. Variations on the Stochastic Shortest Path Problem. In Deepak D'Souza, Akash Lal, and Kim Guldstrand Larsen, editors, *Verification, Model Checking, and Abstract Interpretation - 16th International Conference, VMCAI 2015, Mumbai, India, January 12-14, 2015. Proceedings*, volume 8931 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2015. doi:10.1007/978-3-662-46081-8_1.

37 Mickael Randour, Jean-François Raskin, and Ocan Sankur. Percentile queries in multi-dimensional Markov decision processes. *Formal Methods in System Design*, 50(2-3):207–248, 2017. doi:10.1007/s10703-016-0262-7.

38 Stéphane Le Roux, Arno Pauly, and Mickael Randour. Extending Finite-Memory Determinacy by Boolean Combination of Winning Conditions. In Sumit Ganguly and Paritosh K. Pandya, editors, *38th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2018, December 11-13, 2018, Ahmedabad, India*, volume 122 of *LIPIcs*, pages 38:1–38:20. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018. doi:10.4230/LIPIcs.FSTTCS.2018.38.

39 Stephen D. Travers. The complexity of membership problems for circuits over sets of integers. *Theor. Comput. Sci.*, 369(1-3):211–229, 2006. doi:10.1016/j.tcs.2006.08.017.

40 Moshe Y. Vardi. Automatic Verification of Probabilistic Concurrent Finite-State Programs. In *Proc. of FOCS*, pages 327–338. IEEE, 1985.