Abstract

In the window mean-payoff objective, given an infinite path, instead of considering a long run average, we consider the minimum payoff that can be ensured at every position of the path over a finite window that slides over the entire path. In [7], the problem to decide if in a two-player game, Player 1 has a strategy to ensure a window mean-payoff of at least 0, has been studied.

In this work, we consider a function that given a path returns the supremum value of the window mean-payoff that can be ensured over the path and we show how to compute its expected value in Markov chains and Markov decision processes. We consider two variants of the function: Fixed window mean-payoff in which a fixed window length $l_{\text{max}}$ is provided; and Bounded window mean-payoff in which we compute the maximum possible value of the window mean-payoff over all possible window lengths. Further, for both variants, we consider (i) a direct version of the problem where for each path, the payoff that can be ensured from its very beginning and (ii) a non-direct version that is the prefix independent counterpart of the direct version of the problem.

In the case of Markov chains, we show that computing the expected value of the fixed window mean-payoff is polynomial in the size of the Markov chain and in the window size $l_{\text{max}}$, that is, when $l_{\text{max}}$ is given in unary, while the bounded window mean-payoff problem is polynomial in the size of the Markov chain. In the case of MDPs, computing an optimal strategy that maximizes the expected value of the non-direct fixed window mean-payoff is polynomial in the size of the MDP and the window size $l_{\text{max}}$, while the direct version of the problem is polynomial in the size of the MDP and exponential in $l_{\text{max}}$. As a lower bound, for the non-direct fixed objective, we show that the problem is PSPACE-hard even when $l_{\text{max}}$ is given in unary implying that one cannot have an algorithm that is polynomial in $l_{\text{max}}$ unless P=PSPACE. For bounded window mean-payoff problem, we show that computing the optimal strategy to maximize the expected value of the bounded window mean-payoff is in NP ∩ coNP and is at least as hard as two-player mean-payoff games. Finally, we show that the optimal value of the expected bounded window mean-payoff in an MDP equals the supremum of the fixed window mean-payoff over all $l_{\text{max}}$ and all strategies.

1 Introduction

Markov Decision processes (MDPs) are a classical model for decision-making inside stochastic environments [14, 1]. In this model, a stochastic model of the environment is formalized and we aim at finding strategies that maximize the expected performance of the system with that stochastic environment. This performance in turn is formalized by a function that maps each
infinite path in the MDP to a value. One classical such function is the mean-payoff function that maps an infinite path to the limit of the means of the payoffs obtained on its prefixes. While this measure is classical, alternatives to the mean-payoff measure have been studied in the literature, e.g. one of the most studied alternative notion is the notion of discounted sum \[14\]. The main drawback of the mean-payoff value is that it does not guarantee local stability of the values along the path: if the limit mean-value of an infinite path converges to value \(v\), it may be the case that for arbitrarily long infixes of the infinite path, the mean-payoff of the infix is largely away from this value. There have been several recent contributions \[6, 3, 7, 4\] in the literature to deal with this possible fluctuations from the mean-payoff value along a path. In this paper, we study the notion of window mean-payoff that was introduced in \[6, 7\] for two player games, in the context of MDPs, and we provide algorithms and prove computational complexities for the expected value of window mean-payoff objectives.

As introduced in \[7\], in a window mean-payoff objective instead of the limit of the mean-payoffs along the whole sequence of payoffs, we consider payoffs over a local finite length window sliding along the infinite sequence: the objective asks that the mean-payoff must always reach a given threshold within the window length \(l_{\text{max}}\). This objective is clearly a strengthening of the mean-payoff objective: for all lengths \(l_{\text{max}}\), and all infinite sequences \(\pi\), payoffs that satisfy the window mean-payoff objective for threshold \(\lambda\) imply that \(\pi\) has a mean-payoff value of at least \(\lambda\). It can also be shown that this additional stability property can always be met at the cost of a small degradation of mean-payoff performances in two-player games: whenever there exists a strategy that forces a mean-payoff value \(v\) against any behavior of the adversary then for every \(\epsilon > 0\), there is a window length \(l_{\text{max}}\) and a strategy that ensure that the window mean-payoff objective for threshold \(v - \epsilon\) is eventually satisfied for windows of length \(l_{\text{max}}\) (see Proposition 1 in \[12\]).

In this paper, we study how to maximize the expected value of the window mean-payoff function \(f_{l_{\text{max}} WMP}\) in MDPs. The value of an infinite sequence of integer values \(\pi : \mathbb{N} \rightarrow \mathbb{Z}\) for this function is defined as follows:

\[
f_{l_{\text{max}} WMP}(\pi) = \sup \{ \lambda \in \mathbb{R} \mid \forall i \in \mathbb{N} : \lambda = \max_{1 \leq j \leq i} \frac{1}{j} \sum_{k=0}^{j-1} \pi(i + k) \geq \lambda \}
\]  

i.e., it returns the supremum of all window mean-payoff thresholds that are enforced by the sequence of payoffs \(\pi\) over the window length \(l_{\text{max}}\). As in \[12\], we study natural variants of this measure: (i) when the length of the window is fixed or it is left unspecified but needs to be bounded, and (ii) when the window property needs to be true from the beginning of the path, or a prefix independent variant which asks the window property to eventually hold from some position in the path (leading to a prefix independent variant).

**Main contributions** Our results are as follows. First, for the prefix independent version of the measure \(f_{l_{\text{max}} WMP}\) and for a fixed window length \(l_{\text{max}}\), we provide an algorithm to compute the best expected value of \(f_{l_{\text{max}} WMP}\) with a time complexity that is polynomial in the size of the MDP \(M\) and in \(l_{\text{max}}\) (Theorem 15). It is worth to note that, since the main motivation for introducing window mean-payoff objective is to ensure strong stability over reasonable period of time, it is very natural to assume that \(l_{\text{max}}\) is bounded polynomially by the size of the the MDP \(M\). This in turn implies that our algorithm is fully polynomial for the most interesting cases. We also note that this complexity matches the complexity of computing the value of the function \(f_{l_{\text{max}} WMP}\) for two-player games \[7\], and we provide a relative hardness result: the problem of deciding the existence of a winning strategy in a window mean-payoff game can be reduced to the problem of deciding if the maximal expected mean-payoff value of a MDP for \(f_{l_{\text{max}} WMP}\) is
larger than or equal to a given threshold \( \lambda \) (Theorem 16).

Second, we consider the case where the length \( l_{\text{max}} \) in the measure \( f_{\text{WMP}}^{l_{\text{max}}} \) is not fixed but only required to be bounded. In that case, we provide an algorithm which is in \( \text{NP} \cap \text{coNP} \) (Theorem 21). In addition, we show that providing a polynomial time solution to our problem would also provide a polynomial time solution to the value problem in mean-payoff games (Theorem 22), this is a long-standing open problem in the area [16].

Third, we consider the case where the good window property needs to be imposed from the start of the path. In that case, surprisingly, the problem of computing if there is a strategy to obtain an expected value above a threshold \( \lambda \) is provably harder than for two-player games unless \( \text{P}=\text{PSpace} \). Indeed, while the threshold problem for the worst-case value can be solved in time polynomial in the size of the game and in \( l_{\text{max}} \), we show that for the expected value in a MDP, the problem is \( \text{PSPACE-Hard} \) even if \( l_{\text{max}} \) is given in unary (Theorem 27). To solve the problem, we provide an algorithm that executes in time that is polynomial in the size of the MDP, polynomial in the largest payoff appearing in the MDP, and exponential in the length \( l_{\text{max}} \) (Theorem 26).

Finally, while our main results concentrate on MDP, we also systematically provide results for the special case of Markov chains.

Related Works As already mentioned, the window mean-payoff objective was introduced in [6] for two-player games. We show in this paper that the complexity of computing maximal expected value for the window mean-payoff value function is closely related to the computation of the worst-case value of a game inside end-components of the MDP (see Theorem 12 and 19) for the prefix independent version of our objective. For the non-prefix independent version, surprisingly, computing the expected value for MDP seems computationally more demanding than computing the mean-payoff value for games (unless \( \text{P}=\text{PSpace} \)). The window mean-payoff objectives were also considered in games with imperfect information in [12], and in combination with omega-regular constraints in [5].

Stability issues of the mean-payoff measure have been studied in several contributions. In [3], the authors study MDP where the objective is to optimize the expected mean-payoff performance and stability. They propose alternative definitions to the classical notions of statistical variance. The notion of stability that is offered by window mean-payoff objective and that has been studied in this paper is stronger than the one proposed in [3]. The techniques needed to solve the two problems are very different too as they mainly rely on solving sets of quadratic constraints.

In [4], window-stability objectives have been introduced. Those objectives are inspired from the window mean-payoff objective of [3] but they are different in that they do not enjoy the so called inductive window property because of the stricter stability constraints that those objectives impose. The authors have considered the window-stability objectives in the context of games (2 players) and graphs (1 player) but they did not consider the case of MDP (1 \( \frac{1}{2} \) players).

MDP with classical mean-payoff objectives have been extensively studied both for the probabilistic threshold and the expectation payoff problem, see e.g. [14]. Combination of both type of constraints have been considered in [2].

Structure of the paper Section 2 introduces the definitions and the formal concepts used in this paper. In Section 3 we study the expected window mean-payoff problems in weighted Markov chains, while in Section 4 we study the problems for weighted Markov decision processes. We give algorithms to solve the problems, as well as we give hardness results for the problems we study. In Section 5 we show that in an MDP, the value of the bounded window
mean-payoff problem equals the supremum of the fixed window mean-payoff problem over all window lengths and over all strategies. Section 6 defines window mean-cost instead of window mean-payoff where the objective is to minimize the cost as opposed to maximizing payoff. Finally, we conclude in Section 7 with a summary of the complexity and the hardness results that we obtain for both Markov chains and MDPs. Some details related to some of the proofs and an additional algorithms for the direct fixed window mean-payoff problem in Markov chains have been moved to the appendix.

2 Preliminaries

2.1 Weighted Markov Chains

Given a finite set $A$, a (rational) probability distribution over $A$ is a function $Pr : A \rightarrow [0,1] \cap \mathbb{Q}$ such that $\sum_{a \in A} Pr(a) = 1$. We denote the set of probability distributions over $A$ by $\mathcal{D}(A)$.

A finite weighted Markov chain (MC, for short) is a tuple $\mathcal{M} = (S, E, s_{\text{init}}, w, \mathbb{P})$, where $S$ is the finite set of states, $s_{\text{init}} \in S$ is the initial state, $E \subseteq S \times S$ is the set of edges, the function $w : E \rightarrow \mathbb{Q}$ defines the weights of the edges, and $\mathbb{P} : S \rightarrow \mathcal{D}(E)$ is a function that assigns a probability distribution – on the set $E(s)$ of outgoing edges from $s$ – to all states $s \in S$. A Markov chain is infinite if the set $S$ is not finite. Unless stated otherwise, we always consider a Markov chain to be finite.

Given a state $s \in S$, we define the support of $s$ to be $\text{Supp}(s) = \{s' \in S \mid (s,s') \in E\}$, that is, every edge has a non-zero probability. Then, for a state $s \in S$, we define the set of infinite paths in $\mathcal{M}$ starting from $s$ as $\text{Paths}^\mathcal{M}(s) = \{\pi = s_0s_1 \ldots s_n \in S^\infty \mid s_0 = s, \forall n \in \mathbb{N}, s_{n+1} \in \text{Supp}(s_n)\}$, and set of infinite paths in the whole Markov chain is denoted $\text{Paths}^\mathcal{M}$ and is equal to $\bigcup_{s \in S} \text{Paths}^\mathcal{M}(s)$. Then, the set of finite paths from a state $s$ is denoted $\text{FPaths}^\mathcal{M}(s)$ and corresponds to the set of all prefixes of all the paths in $\text{Paths}^\mathcal{M}(s)$. Similarly, the set of finite paths in the whole Markov chain $\text{FPaths}^\mathcal{M}$ is equal to $\bigcup_{s \in S} \text{FPaths}^\mathcal{M}(s)$. We denote by $\text{Last}(\rho)$ the last vertex $s_n$ of the finite path $\rho = s_0 \ldots s_n$. Finally, the set of paths of length $l \geq 1$ starting from $s$ can be defined as $\text{FPaths}^l_\mathcal{M}(s) = \{\pi \in \text{FPaths}^\mathcal{M}(s) \mid |\pi| = l\}$, where $|\pi|$ denotes the number of edges in $\pi$. The set of finite paths of length $l$ in $\mathcal{M}$, that is $\text{FPaths}^l_{\text{max}}$, is equal to $\bigcup_{s \in S} \text{FPaths}^l_{\text{max}}(s)$.

If we consider a weighted Markov chain $\mathcal{M} = (S, E, s_{\text{init}}, w, \mathbb{P})$ and a path $\rho \in \text{FPaths}^\mathcal{M}$, we denote by $\text{Cyl}(\rho)$ the cylinder set generated by $\rho$ and it is defined as $\text{Cyl}(\rho) = \{\pi \in \text{Paths}^\mathcal{M} \mid \rho \text{ is a prefix of } \pi\}$. The interesting property of these cylinder sets is that, for any path $\rho \in \text{FPaths}^\mathcal{M}$, we have: $\text{Pr}(\text{Cyl}(\rho)) = \mathbb{P}(\rho)$.

The bottom strongly connected components (BSCCs for short) in a Markov chain $\mathcal{M}$ are the strongly connected components from which it is impossible to exit. That is, a BSCC $\mathcal{B}$ ensures that $\forall s \in \mathcal{B}$, $\text{Supp}(s) \subseteq \mathcal{B}$. We denote by $\text{BSCC}(\mathcal{M})$ the set of BSCCs of the Markov chain $\mathcal{M}$. A nice property of these components is that every infinite path $\pi \in \text{Paths}^\mathcal{M}$ eventually ends up in one of the BSCCs almost surely. Formally, we have the proposition:

**Proposition 1.** For all state $s \in S$, we have: $\text{Pr}(\pi \in \text{Paths}^\mathcal{M}(s) \mid \exists \mathcal{B} \in \text{BSCC}(\mathcal{M}), \pi \models \Diamond (\Box \mathcal{B}) = 1$.

To compute the different expected values we are interested in, we will need some computations on Markov chains. More specifically, given a weighted Markov chain $\mathcal{M} = (S, E, s_{\text{init}}, w, \mathbb{P})$, we would like to compute $Pr_s(\{\pi \mid \pi \models \Diamond B\})$ or $Pr_s(\{\pi \mid \pi \models \text{CUB}\})$ from a state $s \in S$, where $B$ and $C$ are two sets of states, $\Diamond B$ denotes eventually reaching some state in $B$ and $\text{CUB}$ denotes until some state in $B$ is reached, only states in $C$ are visited. In the following,
A sub-MDP of $\Gamma$ is called a sub-MDP. A sub-MDP $(T, A)$ is a pair $(P_{\text{ost}}T, A)$ almost surely, whatever strategy is in the set of outgoing edges from $s$ to all states $s \in S$ if action $a \in A$ is taken in $s$. Given a state $s \in S$, we denote by $\text{Act}(s)$ the set of actions $\{a \in \text{Act} \mid \exists s' \in S, P(s, a, s') > 0\}$. Then, given an action $a \in \text{Act}(s)$, we define $\text{Post}(s, a) = \{s' \in S \mid P(s, a, s') > 0\}$. We assume that, for all $s \in S$, $\text{Act}(s) \neq \emptyset$.

A strategy in $\Gamma$ is a function $\sigma : S^+ \rightarrow \text{Act}$ such that $\sigma(s_0 \ldots s_n) \in \text{Act}(s_n)$, for all $s_0 \ldots s_n \in S^+$. A finite-memory strategy can be seen as tuple $(M, \text{act}, \delta, \text{start})$ where:

- $M$ is a set of modes;
- $\text{act} : M \times S \rightarrow \text{Act}$ selects an action such that, for all $m \in M$ and $s \in S$, $\text{act}(m, s) \in \text{Act}(s)$;
- $\delta : M \times S \rightarrow M$ is a mode update function;
- $\text{start} : S \rightarrow M$ selects the initial mode for any state $s \in S$.

The amount of memory used by such a strategy is defined to be $|M|$. Once we fix a strategy in an MDP, we obtain an MC. If the strategy is not finite memory, we may obtain an infinite Markov chain. For $\sigma \in \text{strat}(\Gamma)$, we denote by $\Gamma^{[\sigma]}$ the Markov chain that we obtain by applying strategy $\sigma$ to $\Gamma$. A strategy is said to be memoryless if $|M| = 1$, that is, the choice of action only depends on the current state where the choice is made. Formally, a strategy is said to be memoryless if for all finite paths $\rho_1$ and $\rho_2$ in $\text{FPaths}^{[\sigma]}$ such that $\text{Last}(\rho_1) = \text{Last}(\rho_2)$, we have $\sigma(\rho_1) = \sigma(\rho_2)$. We denote by $\text{strat}(\Gamma)$ the set of all strategies, and by $\text{strat}_{\text{f}}(\Gamma)$ the set of all memoryless strategies. Note that this set is finite.

We now define maximal end components in an MDP. To do this, we first define the notion of sub-MDP and the directed graph induced by it. For an MDP $\Gamma = (S, E, \text{Act}, s_{\text{init}}, w, P)$, a pair $(T, A)$ where $\emptyset \neq T \subseteq S$ and $A : T \rightarrow 2^{\text{Act}}$ such that:

- $\emptyset \neq A(s) \subseteq \text{Act}(s)$ for all state $s \in S$;
- for $s \in T$, $a \in A(s)$, we have $\text{Post}(s, a) \subseteq T$;

is called a sub-MDP of $\Gamma$. A sub-MDP $(T, A)$ induces a directed graph $G_{(T, A)}$ whose vertex are $T \cup \{(s, a) \in T \times \text{Act}(s)\}$ and whose edges are $\{(s, (s, a)) \mid s \in T, a \in A(s)\} \cup \{((s, a), t) \mid t \in \text{Post}(s, a)\}$. Then, an end-component is a sub-MDP whose induced graph is strongly connected. Finally, a maximal end component (MEC, for short) is an end component that is included in no other end component. We denote by $\text{MEC}(\Gamma)$ the set of all maximal end components of $\Gamma$. Computing the set $\text{MEC}(\Gamma)$ can be done in time $O(|S| \cdot (|S| + |E|))$ (see [II]). Any infinite path will eventually end up in one maximal end component almost surely, whatever strategy is considered. That is, we have the following proposition:

**Proposition 2.** For all strategy $\sigma \in \text{strat}(\Gamma)$, for all state $s \in T$, we have: $Pr(\pi \in \text{Paths}^{[\sigma]}(s) \mid \exists M \in \text{MEC}(\Gamma), \pi \models \diamond \Box M) = 1$. 

5
In several cases, we will need to compute the (optimal) expected value of the mean-payoff in an MDP $\Gamma = (S, E, Act, s_{init}, w, P)$. This can be done in polynomial time (see [14]). In the following, we will denote the complexity of computing the optimal expected value in $\Gamma$ by $\text{Avg}(|S|, |Act|)$.

The size of the MDP $\Gamma$, that is denoted $|\Gamma|$, corresponds to the value $|S| + |E| + \log(W)$ where $W = \max_{e \in E} w(e)$.

### 2.3 Weighted Two-Player Games

We introduce here the notion of a finite weighted two-player game.

A finite two-player game is a weighted graph $G = (S_1, S_2, s_{init}, E, w)$ where the set of vertices $S = S_1 \uplus S_2$ is partitioned into the vertices belonging to Player 1, that is $S_1$, and the vertices belonging to Player 2, that is $S_2$, and $s_{init} \in S_1$ is the initial vertex. The set of edges $E \subseteq S \times S$ is such that for all $s \in S$, there exists $s' \in S$ such that $(s, s') \in E$. The weight function $w$ is such that $w : E \to \mathbb{Q}$.

The strategies in a two-player game are analogous to the strategies in an MDP, except that Player 1 chooses an edge from the states in $S_1$, while Player 2 chooses an edge from the states in $S_2$ (instead of choosing an action, that leads to a probabilistic distribution over the edges as in an MDP). We denote by $\text{strat}_1(G) = \{ \sigma : S^* S_1 \to E \}$ and $\text{strat}_2(G) = \{ \sigma : S^* S_2 \to E \}$ the set of all strategies of Player 1 and 2 respectively available in the game $G$.

Given an MDP $\Gamma = (S, E, Act, s_{init}, w, P)$, we denote by $G_{\Gamma}$ the two-player game $(S_1, S_2, s_{init}, E', w')$ resulting from $\Gamma$ where:

- $S_1 = S$;
- $S_2 = S \times Act$;
- $E' = \{(s, (s, a)) \in S_1 \times S_2 \mid s \in S, a \in Act(s) \} \cup \{((s, a), s') \in S_2 \times S_1, P(s, a, s') > 0 \}$;
- $w'(e) = 0$ if $e \in E_1$;
- $w'(e) = w(e)$ if $e \in E_2$.

It has to be noted that any strategy $\sigma_1$ of Player 1 in $G_{\Gamma}$ can be seen as a strategy in the MDP $\Gamma$. Reciprocally, any strategy $\sigma$ in the MDP $\Gamma$ can be seen as a strategy for Player 1 in $G_{\Gamma}$. Therefore, we have $\text{strat}(\Gamma) = \text{strat}_1(G_{\Gamma})$.

In a two-player game, there is no randomness. Therefore, once the deterministic strategies are fixed, there is a unique path that can occur in the game, from the initial state $s$. For a game $G$, given two strategies $\sigma_1 \in \text{strat}_1(G)$ and $\sigma_2 \in \text{strat}_2(G)$, we denote by $\pi_{(G, s, \sigma_1, \sigma_2)}$ the path that occur in the two-player game $G$ under strategies $\sigma_1$ and $\sigma_2$ from state $s$. Then, if we consider a function $f$ that associate a value to any infinite path of a two-player game, we denote by $V_s^f(G, \sigma_1, \sigma_2)$ the value $f(\pi_{(G, s, \sigma_1, \sigma_2)})$.

In the two-player game $G_{\Gamma}$ resulting from an MDP $\Gamma$, the set of paths from a state $s$ that may occur when Player 1 chooses a strategy $\sigma_1 \in G_{\Gamma}$ from $s$ is defined as $\text{Paths}^{G, \sigma_1}(s) = \{\pi_{(G, s, \sigma_1, \sigma_2)} \mid \sigma_2 \in \text{strat}_2(G)\}$. Then, we may consider the function $p_s^{\sigma_1} : \text{Paths}^{G, \sigma_1}(s) \to \text{Paths}^{G, \sigma_1}(s)$ such that for $\pi = s_0s_1 \ldots \in \text{Paths}^{G, \sigma_1}(s)$ where $s_0 = s$, we have $p_s^{\sigma_1}(\pi) = t_0t_1 \ldots \in \text{Paths}^{G, \sigma_1}(s)$ with:

- $t_0 = s_0 \in S_1$;
- $\forall i \geq 1,$
For a Markov chain $\mathcal{M} = (S, E, s_{\text{init}}, w, \mathbb{P})$, a path $\pi = s_0 s_1 \ldots \in \text{Paths}^\mathcal{M}$ and $i, l \in \mathbb{N}$, by $\pi(i \ldots (i + l))$ we refer to the sequence of $l + 1$ states $s_i s_{i+1} \ldots s_{i+l}$ (that is also a sequence of $l$ edges) and by $\mathbb{P}(\pi(i \ldots (i + l)))$ we refer to $\prod_{j=0}^{l-1} \mathbb{P}(s_i, s_{i+1})$. Consider some measurable function $f : \text{Paths}^\mathcal{M}(s_{\text{init}}) \to \mathbb{R}$ associating a value to each infinite path starting from $s_{\text{init}}$. For an interval $I \subset \mathbb{R}$, we denote by $f^{-1}(\mathcal{M}, s_{\text{init}}, I)$ the set $\{ \pi \in \text{Paths}^\mathcal{M}(s_{\text{init}}) \mid f(\pi) \in I \}$, and for $r \in \mathbb{R}$, $f^{-1}(\mathcal{M}, s_{\text{init}}, r)$ refers to $f^{-1}(\mathcal{M}, s_{\text{init}}, [r, r])$. Since the set of paths $\text{Paths}^\mathcal{M}(s_{\text{init}})$ forms a probability space, and $f$ is a random variable, we denote by $\mathbb{E}^\mathcal{M}_{s_{\text{init}}}(f)$ the expected value of $f$ over the set of paths starting from $s_{\text{init}}$.

Finally, for $k \in \mathbb{N}$, we denote respectively by $[k]_0$ and $[k]$ the set of natural numbers $\{0, \ldots, k\}$ and $\{1, \ldots, k\}$ respectively.

### 2.4 Further Notations

Consider a set $E$, a payoff function $w : E \to \mathbb{Q}$, and $\rho = e_0 \ldots e_{l-1} \in E^l$. We first define the function $\text{WTP}$ (that stands for $\text{Window Total-Payoff}$) such that:

$$\text{WTP}(\rho, l) = \max_{k \in [l]} \sum_{i=1}^{k} w(e_i)$$

The value $\text{WTP}(\rho)$ is the maximum total payoff one can ensure over a window of length $k \in [l]$ starting from the initial vertex of the sequence of edges.

Similarly, we define $\text{WMP}$ (that stands for $\text{Window Mean-Payoff}$) such that:

$$\text{WMP}(\rho, l) = \max_{k \in [l]} \frac{1}{k} \sum_{i=1}^{k} w(e_i)$$

The value $\text{WMP}(\rho)$ is the maximum mean-payoff one can ensure over a window of length $k \in [l]$. For a given infinite path $\pi$, a threshold $\lambda \in \mathbb{Q}$, a position $i \in \mathbb{N}$ and $l \in [l_{\text{max}}]$, we say that the window $\pi(i \ldots (i + l))$ is closed if $\text{WMP}(\pi(i \ldots (i + l))) \geq \lambda$. Otherwise, the window is open. We note that $\text{WMP}(\rho, l') \geq \text{WMP}(\rho, l)$ for $l' \geq l$.

Given a Markov chain $\mathcal{M}$ with an initial state $s_{\text{init}}$ and a rational threshold $\lambda \in \mathbb{Q}$, we define the following objectives.

- Given $l_{\text{max}} \in \mathbb{N}$, the good window objective

$$\text{GW}(\lambda, l_{\text{max}}) = \left\{ \pi \in \text{Paths}^\mathcal{M}(s_{\text{init}}) \mid \lambda \leq \text{WMP}(\pi(0 \ldots l_{\text{max}}), l) \right\}.$$  \hspace{1cm}(2)

requires that there exists a window starting in the first position of $\pi$ and of size at most $l_{\text{max}}$ over which the mean-payoff is bounded below by the threshold $\lambda$. Again, $\text{GW}(\lambda, l'_{\text{max}}) \supseteq \text{GW}(\lambda, l_{\text{max}})$ for $l'_{\text{max}} \geq l_{\text{max}}$. 

7
• Given $l_{\text{max}} \in \mathbb{N}$, the \textit{direct fixed window mean-payoff} objective

$$\text{DirFixWMP}(\lambda, l_{\text{max}}) = \left\{ \pi \in \text{Paths}^M(s_{\text{init}}) \mid \forall j \geq 0, \pi(j, \infty) \in \text{GW}(\lambda, l_{\text{max}}) \right\}$$

(3)

requires that good windows of size at most $l_{\text{max}}$ exist in all positions along the path.

• The \textit{direct bounded window mean-payoff} objective

$$\text{DirBWMP}(\lambda) = \left\{ \pi \in \text{Paths}^M(s_{\text{init}}) \mid \exists l_{\text{max}} > 0, \pi \in \text{DirFixWMP}(\lambda, l_{\text{max}}) \right\}$$

(4)

requires that there exists a bound $l_{\text{max}}$ such that the path satisfies the direct fixed objective for the length $l_{\text{max}}$.

• Given $l_{\text{max}} \in \mathbb{N}$, the \textit{fixed window mean-payoff} objective

$$\text{FixWMP}(\lambda, l_{\text{max}}) = \left\{ \pi \in \text{Paths}^M(s_{\text{init}}) \mid \exists i \in \mathbb{N}, \pi(i, \infty) \in \text{DirFixWMP}(\lambda, l_{\text{max}}) \right\}$$

(5)

is the \textit{prefix-independent} version of the direct fixed window objective: it requires for the existence of a suffix of the path satisfying it.

• The \textit{bounded window mean-payoff} objective

$$\text{BWMP}(\lambda) = \left\{ \pi \in \text{Paths}^M(s_{\text{init}}) \mid \exists l_{\text{max}} > 0, \pi \in \text{FixWMP}(\lambda, l_{\text{max}}) \right\}$$

(6)

is the prefix-independent version of the direct bounded window objective.

2.6 Functions of Interest

For each of those objective, we associate a value to every infinite path. We define the following functions, respectively for the \textit{fixed}, \textit{direct fixed}, \textit{bounded} and \textit{direct bounded window mean-payoff} problem:

$$f_{\lambda \text{FixWMP}}^{l_{\text{max}}} (\pi) = \sup \{ \lambda \in \mathbb{Q} \mid \pi \in \text{FixWMP}(\lambda, l_{\text{max}}) \} $$

$$f_{\lambda \text{DirFixWMP}}^{l_{\text{max}}} (\pi) = \sup \{ \lambda \in \mathbb{Q} \mid \pi \in \text{DirFixWMP}(\lambda, l_{\text{max}}) \} $$

$$f_{\lambda \text{DirBWMP}} (\pi) = \sup \{ \lambda \in \mathbb{Q} \mid \pi \in \text{DirBWMP}(\lambda) \} $$

$$f_{\lambda \text{BWMP}} (\pi) = \sup \{ \lambda \in \mathbb{Q} \mid \pi \in \text{BWMP}(\lambda) \} $$

It is now possible to study the complexity of finding the expected value of these functions in a Markov chain and in a Markov decision process. In the following, in the MCs and MDPs w.l.o.g. we consider only non-negative integer weights. Note that if the weights belong to $\mathbb{Q}$, then one can multiply them with the LCM of their denominators to obtain integer weights. Among the resultant set of integer weights, if the minimum integer weight $\kappa$ is negative, then we add $-\kappa$ to the weight of each edge so as to obtain weights that are natural numbers.
2.7 A First Result

The first observation is that the functions \( f_{BWMP} \) and \( f_{DirBWMP} \) always produce the same result. That is stated in the following theorem.

**Lemma 3.** Let \( M = \langle S, E, s_{init}, w, \mathcal{F} \rangle \) be a Markov chain and let \( \pi = s_0 \ldots \in \text{Paths}^M \). Then:

\[
f_{DirBWMP}(\pi) = f_{BWMP}(\pi)
\]

It is easy to see that \( f_{DirBWMP}(\pi) \leq f_{BWMP}(\pi) \). Now for every \( \epsilon \in \mathbb{R}_{\geq 0} \), a window mean-payoff value of \( f_{BWMP}(\pi) - \epsilon \) can be ensured from the beginning of the path \( \pi \) by considering appropriately large window length. Since \( f_{DirBWMP}(\pi) \) has been defined as the supremum of the window mean-payoff values that can be ensured along the path \( \pi \) with arbitrarily large window lengths, the result follows. The detailed proof is given in Appendix B.

3 Expected Window Mean-payoff in Markov Chains

In this section, we study the fixed, direct fixed and bounded window mean-payoff problems on weighted Markov chains. We show that while the bounded window mean-payoff problem can be solved in polynomial time, the algorithm for the fixed window mean-payoff problem is polynomial in the value of the window length, that is, it is polynomial when the window length is given in unary, and pseudopolynomial if it is given in binary. Then, the algorithm for the direct fixed window mean-payoff problem is polynomial in the value of the window length and the weights appearing in the Markov chains. Therefore, it is pseudopolynomial when they are given in binary.

3.1 Fixed Window Mean-Payoff

We are interested in the expected value of \( f_{fixWMp}^{l_{max}} \) for a given \( l_{max} \) in a Markov chain. More specifically, consider a Markov chain \( M \) and an initial state \( s_{init} \). Then, we study

\[
E_{s_{init}}^{M}(f_{fixWMp}^{l_{max}}) = \int_{r \in \mathbb{Q}} r \cdot Pr((f_{fixWMp}^{l_{max}})^{-1}(M, s_{init}, r))
\]

In order to compute this value, first consider the bottom strongly connected components (BSCCs for short) of the Markov chain \( M \). In the preliminaries, we mentioned that any infinite path will eventually end up in one of them almost surely. Since the fixed window mean-payoff is prefix independent, the value of a path \( \pi \), that is, \( f_{fixWMp}^{l_{max}}(\pi) \), is only determined by the BSCC in which it ends up. Moreover, we have the following result:

**Proposition 4.** Let \( B \) be a BSCC. Let \( \rho = s_0 \ldots s_n \in F\text{Paths}^B \). Then, every path in \( B \) will almost surely visit the sequence of states \( \rho \) infinitely often. Formally:

\[
Pr(\{\pi \in \text{Paths}^B \mid \pi \in \Box \diamond \rho\}) = 1
\]

**Proof.** Consider an infinite path \( \pi \in \text{Paths}^B \). Then, because we are in a BSCC, \( \pi \) will almost surely go infinitely often through \( s_0 \). Moreover, in \( s_0 \), there is a positive probability of taking the path \( \rho \), that is, visiting the states in \( \rho \) in sequence. It follows that \( \pi \) will almost surely reach the sequence of states \( \rho \) infinitely often.

In particular, in a BSCC \( B \in BSCC(M) \), the sequence of \( l_{max} + 1 \) states \( s_0 \ldots s_{l_{max}} \in B \) that minimizes WMP is visited infinitely often by any infinite path almost surely. Hence, the following theorem:
Figure 1: An example of a weighted Markov chain

**Theorem 5.** The expected value of $f_{\text{FixWMP}}^{l_{\text{max}}}$, over paths that are entirely contained in a BSCC does not depend on the starting state of the paths considered. Moreover, if we define the expected value in a BSCC $\mathcal{B}$ as the expected value starting from any state, then we have:

$$
\mathbb{E}^B(f_{\text{FixWMP}}^{l_{\text{max}}}) = \min_{s \in \mathcal{B}} \min_{\pi \in \text{Paths}_{l_{\text{max}}}^M(s)} \text{WMP}(\pi(0 \ldots l_{\text{max}}))
$$

Before proving this theorem, let us look at the example in Figure 1 of a weighted Markov chain to illustrate the definition of $m_B$ for some BSCC $\mathcal{B}$. Such an example can be seen on Figure 1. The probabilities are written next to each edge in black, the weights are in red. Let us consider the case where $l_{\text{max}} = 2$. In this Markov chain, there are two BSCCs: $\mathcal{B}_1 = \{s_1\}$ and $\mathcal{B}_2 = \{s_3, s_4\}$. The value of $m_{\mathcal{B}_1}$ is obvious since there is only one path of length $l_{\text{max}} = 2$ in $\mathcal{B}_1$: $\pi = s_1s_1s_1$ with $\text{WMP}(\pi) = 2$. Let us now consider $m_{\mathcal{B}_2}$. In the BSCC $\mathcal{B}_2$, there are eight paths of length $l_{\text{max}} = 2$.

- From $s_3$:
  - $\pi_3_1 = s_3s_3s_3$, with $\text{WMP}(\pi_3_1) = 3$;
  - $\pi_3_2 = s_3s_3s_4$, with $\text{WMP}(\pi_3_2) = 3$;
  - $\pi_3_3 = s_3s_4s_3$, with $\text{WMP}(\pi_3_3) = 2$;
  - $\pi_3_2 = s_3s_4s_4$, with $\text{WMP}(\pi_3_2) = 2$.

- From $s_4$:
  - $\pi_4_1 = s_4s_3s_3$, with $\text{WMP}(\pi_4_1) = 1.5$;
  - $\pi_4_2 = s_4s_3s_4$, with $\text{WMP}(\pi_4_2) = 1$;
  - $\pi_4_3 = s_4s_4s_3$, with $\text{WMP}(\pi_4_3) = 1$;
  - $\pi_4_4 = s_4s_4s_4$, with $\text{WMP}(\pi_4_4) = 1$.

The minimum over these paths is achieved for paths $\pi_4_2$, $\pi_4_3$ and $\pi_4_4$. Therefore, $m_{\mathcal{B}_2} = \text{WMP}(\pi_4_1) = 1$. Moreover, since both BSCCs have the same probability (0.5) of being reached, by applying Formula 7, we obtain that the expected value of the fixed window mean-payoff in this Markov chain $\mathcal{M}$ is equal to $\mathbb{E}^M(f_{\text{FixWMP}}^{l_{\text{max}}}) = 0.5 \cdot 2 + 0.5 \cdot 1 = 1.5$.

**Proof.** Let $s \in \mathcal{B}$. We prove:

1. For all $p < m_{\mathcal{B}}$, we have $Pr((f_{\text{FixWMP}}^{l_{\text{max}}})^{-1}((\mathcal{M}, s_{\text{init}}, p))) = 0$;
For all $q > m_B$, we have $Pr((f_{FixWMP}^{\max})^{-1}(M, s_{init}, q)) = 0$.

1. Consider a path $\pi \in Path_{s}^{M}(s)$. We prove that $\pi \in FixWMP(m_B, l_{max})$. Let $i \in \mathbb{N}$ and let $m_i = WMP(\pi(i \ldots (i + l_{max})))$. Then, by definition of $m_B$, we have for all $i \in \mathbb{N}, m_i \geq m_B$. Therefore, we have that $\pi \in FixWMP(m_B, l_{max})$. This implies that $f_{FixWMP}^{\max}(\pi) \geq m_B$. Hence, for all $p < m_B$, we have $(f_{FixWMP}^{\max})^{-1}(M, s, p) = \emptyset$ and therefore, $Pr((f_{FixWMP}^{\max})^{-1}(M, s, p)) = 0$.

2. Let us prove the second point. Let us denote by $\rho_{mg}$ the sequence of $l_{max} + 1$ states such that $m_B = WMP(\rho_{mg})$. Because $B$ is a BSCC and according to Proposition 4 we know that $Pr(s \models \square \diamond \rho_{mg}) = 1$, which is equivalent to say that the set $P_B = \{ \pi \in Path(s) \mid \pi \models \diamond \rho_{mg} \}$ of paths that reach infinitely often the sequence of states $\pi_{mg}$ is such that $Pr(P_B) = 1$. Let $\pi \in P_B$. Then $f_{FixWMP}^{\max}(\pi) \leq m_B$. Hence, for all $q > m_B$, we have $(f_{FixWMP}^{\max})^{-1}(M, s, q) \subseteq (P_B)^C$ and therefore, $Pr((f_{FixWMP}^{\max})^{-1}(M, s, q)) \leq Pr((P_B)^C) = 1 - Pr(P_B) = 0$.

Concluding from 1 and 2, we thus have, for all $s \in B$, $\sum_{r \in \mathbb{Q}} r \cdot Pr((f_{FixWMP}^{\max})^{-1}(M, s, r)) = m_B$. Therefore, the expected value in a BSCC $B$ does not depend on the starting state and is always equal to $m_B$. This concludes the proof.

To sum up, the fixed window mean-payoff of a path only depends on where it ends up. Moreover, almost surely, a path end up in a BSCC. Furthermore, the value of any path that ends up in a BSCC $B$ is, almost surely, equal to $m_B$. Thus the expected value of $f_{FixWMP}^{\max}$ in the Markov chain $M$ from the initial state $s_{init}$ is given by:

$$E_{s_{init}}^{M}(f_{FixWMP}^{\max}) = \sum_{B \in \text{BSCC}(M)} Pr(\diamond B) \cdot m_B$$

Let us focus on computing $m_B$ for a BSCC $B$. Ideally, we would compute this value using dynamic programming. More specifically, for $s \in B$ and $l \in [l_{max}]$, we would express $\min_{\pi \in Path_{s}^{B}(s)} WMP(\pi(0 \ldots l))$ as a function of $\min_{\pi \in Path_{s}^{B}(s')} WMP(\pi(0 \ldots l-1))$ for $s' \in Supp(s)$.

However, it seemed not possible in some cases. For instance, consider the BSCC of Figure 2 with $l_{max} = 3$. It is obvious that, from state $s_1$, the best window mean-payoff is 5 by only considering the first edge. And from state $s_0$ the best window mean-payoff is equal to $(2 + 5 + 4)/3 = 11/3$ by considering three edges. Therefore, from state $s_1$, the edge $(s_2, s_3)$ is not interesting since $w(s_2, s_3) = 4 < 5 = w(s_1, s_2)$. But, from state $s_0$, $(s_2, s_3)$ is interesting since $w(s_2, s_3) = 4 > 3.5 = (w(s_0, s_1) + w(s_1, s_2))/2$. The problem comes from the fact that whether the edge $(s_2, s_3)$ is interesting or not depends on the current mean of the path.

Now consider the decision problem where we have to decide if $m_B \geq \lambda$ for some $\lambda \in \mathbb{Q}$. This is equivalent to decide whether or not $m_B \geq 0$ in the BSCC where every weight have been reduced by $\lambda$. Now, an interesting property is that, for every $\pi \in Path_{s_{l_{max}}}^{B}, WMP(\pi(0 \ldots l_{max})) \geq$
Figure 3: An example of a BSCC $B$ where we want to decide if $m_B \geq 0$, for $l_{max} = 3$

0 $\leftrightarrow$ WTP($\pi(0\ldots l_{max})$) $\geq$ 0. Therefore, we can look at the total payoff, instead of looking at the mean-payoff. In that case, whether an edge is interesting to consider or not does not depend on the current sum of the path, it only depends on the value of the edge (if it is positive or negative). For example, let us consider the case where $\lambda = 3$ in the previous BSCC. The BSCC obtained by reducing every weight by 3 can be seen in Figure 3. Now, the edge $(s_2, s_3)$ is interesting from $s_0$ and $s_1$ since $w(s_2, s_3) = 1 > 0$.

The following lemma generalize this result:

**Lemma 6.** Let $B$ be a BSCC.

For $s \in B$ and $l \in [l_{max}]$, we define $TP_l(s) = \min_{\pi \in \text{Paths}^l_M(s)} \text{WTP}(\pi(0\ldots l))$. Then

$$\forall s \in B, \forall l \in [l_{max}], TP_l(s) = \min_{s' \in \text{Supp}(s)} \max_{w(s, s')} \max(w(s, s'), w(s, s') + TP_{l-1}(s'))$$

with $TP_0(s) = 0$ for all $s \in B$.

**Proof.** To prove this lemma, we just use the fact that a path of length $l$ consists of an edge followed by a path of length $l - 1$. More specifically, let $s \in B$ and $l \in [l_{max}]$, then:

$$TP_l(s) = \min_{\pi \in \text{Paths}^l_M(s)} \text{WTP}(\pi(0\ldots l))$$

$$= \min_{s' \in \text{Supp}(s)} \min_{\pi \in \text{Paths}^l_{l-1}(s')} \text{WTP}(\pi(0\ldots l - 1))$$

$$= \min_{s' \in \text{Supp}(s)} \min_{s = s_0\ldots s_{|s|} \in \text{Paths}^l_{l-1}(s')} \max(w(s, s'), \max_{k \in [1..l-1]} w(s, s') + \sum_{i=0}^{k-1} w(s_i, s_{i+1}))$$

$$= \min_{s' \in \text{Supp}(s)} \max(w(s, s'), w(s, s') + \sum_{s = s_0\ldots s_{|s|} \in \text{Paths}^l_{l-1}(s')} \max_{k \in [1..l-1]} w(s_i, s_{i+1}))$$

$$= \min_{s' \in \text{Supp}(s)} \max(w(s, s'), w(s, s') + TP_{l-1}(s'))$$

Algorithm [1] computes and returns the set of states $S_{pos} = \{s \in S \mid TP_{l_{max}}(s) \geq 0\}$ by using Lemma [6]. More precisely, at iteration $i \in [l_{max}]$ of the second for loop, $C(s) = TP_i(s)$.

The number of operations in that algorithm is bounded by $|E| \cdot l_{max}$ since inside the second for loop (line 4-5), every edge is visited exactly once. If $W$ be the maximal weight that appears in the Markov chain, then each operation requires a time linear in $\log(W)$. Thus, the complexity of this algorithm is in $O(|E| \cdot l_{max} \cdot \log(W))$.

The previous algorithm allows us to know if $m_B \geq \lambda$, for some $\lambda \in \mathbb{Q}$. Algorithm [2] uses a binary search between a lower and an upper bound, respectively called $L_B$ and $U_B$ and initially

\footnote{This algorithm is an adaptation of the GoodWin Algorithm from [3] where every state belongs to Player 2.}
This stops whenever $\epsilon$ ensures that, at the end of the binary search, $L_B \leq m_B < U_B$ with $U_B - L_B = \epsilon \leq \frac{1}{l_{max}}$ (this is justified in the following). This can be ensured by calling successively Algorithm 1 where the weights have been reduced by a given $\lambda$. If all states ensure a non negative total payoff, then $\lambda \leq m_B$, otherwise $\lambda > m_B$. Once the binary search is over, we have to find the exact value of $m_B$ between $L_B$ and $U_B$. We know that there exists $(p, q) \in \mathbb{N} \times [l_{max}]$ such that $m_B = \frac{p}{q}$. Moreover, for any $((k, l), (k', l')) \in (\mathbb{N} \times [l_{max}])^2$ such that if $\frac{k}{l} \neq \frac{k'}{l'}$ then $|\frac{k}{l} - \frac{k'}{l'}| = |\frac{k' - k}{ll'}| > \frac{1}{l_{max}} \geq \epsilon$. This implies that if a pair $(p, q) \in (\mathbb{N} \times [l_{max}])^2$ verifying $L_B \leq \frac{p}{q} < U_B$ is found, then necessarily $m_B = \frac{p}{q}$. That is computed in the last for loop (line 15-18).

Algorithm 1 NonNegWindowBSCC($B, l_{max}$)

Require: $B = (S, E, w)$ and $l_{max} \in \mathbb{N}_0$
Ensure: $S_{pos}$ is the set of states that ensure a non-negative window total payoff for any sequence of states of $l_{max}$ starting in them

1: for all $s \in S$ do
2: $C(s) := 0$
3: for all $i \in [l_{max}]$ do
4: for all $s \in S$ do
5: $C(s) := \min_{(s, s') \in E} \max\{w((s, s')), w((s, s')) + C(s')\}$
6: return $S_{pos} := \{s \in S \mid C(s) \geq 0\}$

Algorithm 2 ExpValBSCC($B, l_{max}$)

Require: $B = (S, E, w)$ is a BSCC from a weighted Markov Chain and $l_{max} \in \mathbb{N}_0$
Ensure: $\mu$ is equal to $E_B(f_{FixWMP}^{l_{max}})$

1: $W := \max_{e \in E} |w(e)|$
2: $L_B := 0, U_B := W + 1$
3: $\epsilon := L_B + U_B$
4: while $\epsilon > \frac{1}{l_{max}}$ do
5: $\lambda := \frac{U_B - L_B}{2}$
6: $\epsilon := \frac{\epsilon}{2}$
7: $S' := NonNegWindowBSCC(B = (S, E, w - \lambda))$
8: if $(S == S')$ then
9: $L_B := \lambda$
10: else
11: $U_B := \lambda$
12: for $l \in [l_{max}]$ do
13: $l_B = l \times L_B ; u_B = l \times U_B$
14: if $([l_B] + 1 = [u_B])$ then
15: $\mu := \frac{u_B}{l_B}$
16: return $\mu$

Let us now discuss the complexity of the Algorithm 2. We consider the number $n_{while}$ of times that we enter the while loop. Initially, $\epsilon = W$, and at each step $\epsilon$ is divided by 2. This stops whenever $\epsilon \leq \frac{1}{l_{max}}$. Therefore, $n_{while} \leq \lceil \log(W \cdot l_{max}^2) \rceil$. At each step, the most expensive (in terms of time of computation) operation is the call of Algorithm 1 of complexity
Finally, the last for loop is taking $O(l_{\text{max}} \cdot \log(W))$ time. Therefore, the complexity of Algorithm 2 is in $O(|E| \cdot l_{\text{max}} \cdot \log(W))$.

**Algorithm 3 FixWMP($\mathcal{M}$, $s_{\text{init}}$, $l_{\text{max}}$)**

**Require:** $\mathcal{M} = (S, E, s_{\text{init}}, P, w)$ is a weighted Markov Chain, $s_{\text{init}} \in S$ and $l_{\text{max}} \in \mathbb{N}_0$

**Ensure:** $E$ is equal to $E_{\mathcal{M}}(f_{\text{FixWMP}}(l_{\text{max}}))$

1: $\text{BSCC} \text{ Vect} := BSCC(\mathcal{M})$
2: $E := 0$
3: for $B \in \text{BSCC} \text{ Vect}$ do
   4: $\text{Prob}_B := Pr_{s_{\text{init}}} (\Diamond B)$
   5: $E_B := \text{ExpValBSCC}(B, l_{\text{max}})$
   6: $E += \text{Prob}_B \cdot E_B$
return $E$

Algorithm 3 only consists in applying Formula 7 from the initial state $s_{\text{init}}$. In this algorithm, we first need to find the BSCCs of the Markov chain. One can use the Tarjan algorithm\[15\] to extract the strongly connected components, and then check which of them are BSCCs. This would require $O(|S| + |E|)$ steps. Inside the for loop, computing the probability of reaching a BSCC is done in time $O(\text{Mat}(|S|))$ and computing the expected value of the BSCC consists in calling Algorithm 2 of complexity $O(|E| \cdot l_{\text{max}} \cdot \log(W) \cdot \log(W \times l_{\text{max}}))$. The number of BSCCs in a Markov chain is bounded by $|S|$. Therefore, the complexity of Algorithm 3 is in $O(|S| \cdot (\text{Mat}(|S|)) + |E| \cdot l_{\text{max}} \cdot \log(W) \cdot \log(W \times l_{\text{max}}))$.

### 3.2 Bounded Window Mean-Payoff

We are interested in the expected value of $f_{BWMP}$ in a Markov chain. More specifically, consider a Markov chain $\mathcal{M} = (S, E, s_{\text{init}}, w, P)$ and an initial state $s_{\text{init}}$. Then, we study

$$\mathbb{E}_{s_{\text{init}}}^\mathcal{M}(f_{BWMP}) = \int_{r \in \mathbb{Q}} r \cdot Pr((f_{BWMP})^{-1}(\mathcal{M}, s_{\text{init}}, r))$$

Since the *bounded window mean-payoff* is prefix independent, the value of a path only depends on the BSCC in which it ends up. Moreover, like in the *fixed window mean-payoff* case, the expected value of $f_{BWMP}$ in a BSCC does not depend on the starting state. We first introduce the function $MP : FPaths^M \to \mathbb{Q}$ such that:

$$\forall \rho = s_0 \ldots s_{|\rho|} \in FPaths^M, \, MP(\rho) = \frac{1}{|\rho|} \sum_{i=0}^{|\rho|-1} w(s_i, s_{i+1})$$

Then, we have the following result:

**Theorem 7.** The expected value of $f_{BWMP}$ in a BSCC $B$ does not depend on the starting state of the paths considered. Moreover, if we denote that expected value by $\mathbb{E}^B(f_{BWMP})$, then we have:

$$\lim_{l \to \infty} \mathbb{E}^B(f_{\text{FixWMP}}^l) = \mathbb{E}^B(f_{BWMP}) = \min_{\rho \in \text{ElemCycle}(B) \text{ not seen twice}} MP(\rho)$$

where $\text{ElemCycle}(B)$ denotes the (finite) set of elementary cycles in $B$, that is the set of cycles composed by states seen only once, except for one which is seen twice.
Proof. From Theorem 5 we have that, for all \( l \geq 1 \), the expected value of \( f_{FizWMP}^l \) inside a BSCC \( B \) does not depend on the starting state and is denoted \( \mathbb{E}^B(f_{FizWMP}^l) \).

First, we prove that the limit of the series \( (\mathbb{E}^B(f_{FizWMP}^l))_{l \geq 1} \) exists:

- \( \forall l \in \mathbb{N}, \mathbb{E}^B(f_{FizWMP}^l) \leq \mathbb{E}^B(f_{FizWMP}^{l+1}) \) since \( \forall \pi \in Paths^B, f_{FizWMP}^l(\pi) \leq f_{FizWMP}^{l+1}(\pi) \);
- \( \forall l \in \mathbb{N}, \mathbb{E}^B(f_{FizWMP}^l) \leq W \), where \( W \) is the maximum over all weights that appear in the Markov chain.

Therefore, according to the monotone convergence theorem, the series \( (\mathbb{E}^B(f_{FizWMP}^l))_{l \geq 1} \) is converging.

Now, let \( s \in B \). We prove that:

1. \( \lim_{l \to \infty} \mathbb{E}^B(f_{FizWMP}^l) \leq \mathbb{E}_s(f_{BWMP}) \);
2. \( \mathbb{E}_s(f_{BWMP}) \leq c_B \);
3. \( c_B \leq \lim_{l \to \infty} \mathbb{E}^B(f_{FizWMP}^l) \).

1. \( \forall l \in \mathbb{N}, \forall \pi \in Paths^B(s), f_{FizWMP}^l(\pi) \leq f_{BWMP}(\pi) \) by definition of \( f_{BWMP} \). Therefore, \( \lim_{l \to \infty} \mathbb{E}^B(f_{FizWMP}^l) \leq \mathbb{E}_s(f_{BWMP}) \).

2. Let \( r > c_B \). Let \( \rho_{cb} = s_0 \ldots s_n \in ElemCycle(B) \) be an elementary cycle (we have \( s_n = s_0 \)) such that \( \text{MP}(\rho_{cb}) = c_B < r \). Consider the smallest index \( i \in \arg \max \sum_{k=0}^{i-1} (w(s_k, s_{k+1}) - c_B) \). The state \( s_i \) is called a high point of the cycle (see, for instance, [7] in the proof of Lemma 10). We prove that the mean-payoff from \( s_i \) to \( s_j \) along the cycle \( \rho_{cb} \) is at most \( c_B \) for all \( j \in [n - 1] \). Recall that \( \rho_{cb} \) is a cycle, therefore \( s_k = s_{k \mod n} \), for all \( k \geq 0 \):

   - If \( i < j \), then \( \sum_{k=0}^{j-1} (w(s_k, s_{k+1}) - c_B) = \sum_{k=0}^{i-1} (w(s_k, s_{k+1}) - c_B) + \sum_{k=i}^{j-1} (w(s_k, s_{k+1}) - c_B) \leq \sum_{k=0}^{i-1} (w(s_k, s_{k+1}) - c_B) \), by definition of \( i \). Therefore, \( \sum_{k=i}^{j-1} (w(s_k, s_{k+1}) - c_B) \leq 0 \);

   - If \( j \leq i \), then \( 0 = \sum_{k=0}^{n-1} (w(s_k, s_{k+1}) - c_B) = \sum_{k=j}^{i-1} (w(s_k, s_{k+1}) - c_B) + \sum_{k=i}^{j-1} (w(s_k, s_{k+1}) - c_B) \geq 0 \), by definition of \( i \).

This is true for all \( j \in [n - 1] \). Then, we denote by \( \rho_{cb}^n(s_i) = s_i \ldots s_{i+n-1} \) the sequence of states that starts from \( s_i \) and goes \( n \) times through the cycle \( \rho_{cb} \). In that case, for every \( n \in \mathbb{N} \), we have:

\[ \forall l \leq n \cdot |\pi_{cb}|, WMP(\rho_{cb}^n(0 \ldots l)) \leq c_B < r \]

Therefore, it implies that for all \( l \geq 1 \), these exists \( \pi \in FPaths^B(s) \) such that \( \forall l_0 \leq l, WMP(\pi(0 \ldots l_0)) < r \). Therefore:

\[ \forall l \geq 1, Pr((f_{FizWMP}^l)^{-1}(\mathcal{B}, s, r)) = 0 \text{ denoted } D_l \]

In fact:
\(- D_1 \subset D_2 \subset D_3 \subset \ldots; \)
\(- \forall l \geq 1, \Pr(D_l) = 0. \)

Therefore:
\[
\Pr\left((f_{BMP})^{-1}(B, s, r)\right) = \Pr\left(\bigcup_{1 \leq l} D_l\right) = \lim_{l \to \infty} \Pr(D_l) = 0
\]

Since this is true for any \( r > c_B \), we have \( \mathbb{E}_B^B(f_{BMP}) \leq c_B. \)

3. Let \( \lambda < c_B. \) We prove that there exists \( l \in \mathbb{N}, \) such that \( \forall \pi \in Paths^B, f_{\text{FixWMP}}^l(\pi) \geq \lambda \) (which implies that \( \mathbb{E}_B^B(f_{\text{FixWMP}}) \geq \lambda \)). We consider the weighted Markov chain restricted to \( B \) where every weight has been subtracted \( \lambda. \) In that case, for all \( \pi \in Paths^B, f_{\text{FixWMP}}^l(\pi) \geq \lambda \) in the original Markov chain is equivalent to \( f_{\text{FixWMP}}^l(\pi) \geq 0 \) in the new Markov chain. We work in that new Markov chain. Let \( d = c_B - \lambda > 0 \) and let \( l = \lceil (|B|-1)W/|B| \rceil \in \mathbb{N} \) where \( W \) is the maximum value of a weight that appears in the new Markov chain.

Let \( \pi \in Paths^B. \) We prove that \( f_{\text{FixWMP}}^l(\pi) \geq 0. \) Let \( i \in \mathbb{N} \) and let \( \pi_i = \pi(i \ldots (i + l)). \) We consider a decomposition of \( \pi_i \) into elementary cycle: \( \pi_i = A, C_1, \ldots, C_n \) constructed as follows: the states of \( \pi_i \) are pushed in a stack, and as soon as a state already seen is pushed in that stack, an elementary cycle, that we remove from the stack, is added to the decomposition. In that decomposition, \( A \) corresponds to the acyclic part of \( \pi_i. \) It has to be noted that the cycles in that decomposition may appear more than once. Consider the function \( \text{Sum} \) that associates to every finite sequences of edges the sum of the weights that appear in that sequence. Because \( A \) constitutes the acyclic part, we have \( |A| \leq |B| - 1. \) Therefore, \( \text{Sum}(A) \geq -(|B| - 1) \cdot W. \) Moreover, the number \( n \) of cycles has to be greater than \( n \geq \frac{l - |B| - 1}{|B|} \geq \frac{|B| - 1 - W}{|B|}, \) since the length of cycle is at most \( |B|. \) Therefore:
\[
\text{Sum}(C_1, \ldots, C_n) = \sum_{i=1}^{n} \text{Sum}(C_i) \geq n \cdot d \geq (|B| - 1) \cdot W
\]

It follows that \( \text{Sum}(\pi_i) \geq 0. \) In fact, \( \forall i \in \mathbb{N}, \text{Sum}(\pi(i \ldots (i + l))) \geq 0, \) that is: \( \forall i \in \mathbb{N}, \text{WMP}(\pi(i \ldots (i + l))) \geq 0. \) Thus, \( f_{\text{FixWMP}}^l(\pi) \geq 0 \) in the new Markov chain.

Therefore, \( \pi \in \text{FixWMP}(\lambda, l) \) in the original Markov chain, and \( f_{\text{FixWMP}}^l(\pi) \geq \lambda \) in the original Markov chain. Since this is true for all \( \pi \in Paths^B, \) we have \( \mathbb{E}_B^B(f_{\text{FixWMP}}) \geq \lambda. \) In fact, \( \forall \lambda < c_B, \exists l \in \mathbb{N}, \mathbb{E}_B^B(f_{\text{FixWMP}}) \geq \lambda. \) That is, \( c_B \leq \lim_{l \to \infty} \mathbb{E}_B^B(f_{\text{FixWMP}}). \)

Thus, the expected value in the BSCC \( B \) starting from any state is equal to the minimum mean-payoff of an elementary cycle in \( B. \)

Hence, in the same way than in the fixed window mean-payoff case, we have:
\[
\mathbb{E}_B^M(f_{BMP}) = \sum_{B \in BSCC(M)} \Pr(\emptyset) \cdot \mathbb{E}_B^B(f_{BMP}) \tag{8}
\]

**Remark 8.** From that formula and Theorem\(^2\) it is possible to show that \( \lim_{l \to \infty} \mathbb{E}_B^M(f_{\text{FixWMP}}) = \mathbb{E}_B^M(f_{BMP}). \)

\(^2\)This decomposition is exactly the same as the one used in \(^2\) to prove Lemma 2
We recall that we are considering a weighted Markov chain \( \mathcal{M} = (S, E, s_{\text{init}}, w, \mathbb{P}) \) from an initial state \( s_{\text{init}} \in S \), with \( W \) the maximum weight appearing in the Markov chain.

Finding the expected value in a BSCC \( \mathcal{B} \) is equivalent to finding the lowest mean-payoff that an elementary cycle can achieve. There exists an algorithm (from Karp, see [13]) that runs in time \( O(|S'| \cdot |E'| \cdot \log(W)) \) that solves this problem, where \( S' \) and \( E' \) are the set of vertices and edges of the BSCC \( \mathcal{B} \).

Algorithm 4 only consists in applying Formula 8 from the initial state \( s_{\text{init}} \). This algorithm is very similar to the one used in the fixed case. In the two algorithms, the only difference lies in Line 5. Computing the BSCCs requires \( O(|S| + |E|) \) steps. Inside the for loop, computing the probability of reaching a BSCC is done in time \( O(\text{Mat}(|S|)) \) and computing the expected value of the BSCC is of complexity \( O(|S| \cdot |E| \cdot \log(W)) \). The number of BSCCs in a Markov chain is bounded by \(|S|\). Therefore, the complexity of Algorithm 4 is in \( O(|S| \cdot (\text{Mat}(|S|) + (|S| \cdot |E| \cdot \log(W)))) \).

```
Algorithm 4 BWMP(\( M, s_{\text{init}} \))

Require: \( M = (S, E, s_{\text{init}}, \mathbb{P}, w) \) is a weighted Markov Chain and \( s_{\text{init}} \in S \)
Ensure: \( E \) is equal to \( E^M_{\text{BWMP}}(f_{\text{BWMP}}) \)
1: BSCC_Vect := BSCC(\( M \))
2: \( E := 0 \)
3: for \( B \in BSCC_Vect \) do
4: \( \text{Prob}_B := Pr_{s_{\text{init}}}(\emptyset|B) \)
5: \( E_B := \text{KarpMeanCycle}(B, l_{\text{max}}) \)
6: \( E = E + \text{Prob}_B \cdot E_B \)
return \( E \)
```

### 3.3 Direct Fixed Window Mean-Payoff

We are interested in the expected value of \( f_{\text{DirFixWMP}}^{l_{\text{max}}} \) for a given \( l_{\text{max}} \) in a Markov chain. More specifically, consider a Markov chain \( \mathcal{M} \) and an initial state \( s_{\text{init}} \). Then, we study

\[
E^M_{s_{\text{init}}}(f_{\text{DirFixWMP}}^{l_{\text{max}}}) = \int_{r \in \mathbb{Q}} r \cdot Pr((f_{\text{DirFixWMP}}^{l_{\text{max}}})^{-1}(\mathcal{M}, s_{\text{init}}, r))
\]

The direct fixed window mean-payoff is not prefix independent, and therefore we cannot only look at the BSCCs. We present here two algorithms to compute that expected value. The first one is polynomial in the size of the Markov chain, in \( l_{\text{max}} \), and in the maximum weight \( W \) that appear in the Markov chain. Thus, the algorithm is pseudopolynomial since \( l_{\text{max}} \) and \( W \) are given in binary. However, it is fixed-parameter tractable when \( l_{\text{max}} \) and \( W \) are considered as parameters. The second one is exponential in \( l_{\text{max}} \), however it is logarithmic in the maximum weight that appears in the Markov chain. Both algorithms consist in building a new Markov chain and then computing probabilities to reach some states in those new Markov chains.

For a given Markov chain \( \mathcal{M} = (S, E, s_{\text{init}}, w, \mathbb{P}) \), an initial state \( s_{\text{init}} \) and a threshold \( \lambda = \frac{\mathcal{A}}{\mathcal{B}} \in \mathbb{Q} \), we would like to compute the probability that the direct fixed window mean-payoff of a path starting in \( s_{\text{init}} \) is at least \( \lambda \), that is \( Pr((f_{\text{DirFixWMP}}^{l_{\text{max}}})^{-1}(\mathcal{M}, s_{\text{init}}, [\lambda, \infty[)) \).

We construct a new Markov chain \( \mathcal{M}_{l_{\text{max}}}^\lambda \) such that the probability that the direct fixed window mean-payoff of a path is at least \( \lambda \) in the original Markov chain \( \mathcal{M} \) is equal to the probability of reaching a specific state (trap) in \( \mathcal{M}_{l_{\text{max}}}^\lambda \).

Let us first define a new weight function \( \bar{w} \) such that:

\[
\forall e \in E, \bar{w}(e) = w(e) \cdot b - a
\]
and the maximum over the new weights:

\[ W = \max_{e \in E} \bar{w}(e) \]

We now define \( \mathcal{M}_{\max}^{\lambda} = \langle S', E', w', s'_{\text{init}}, \mathbb{P}' \rangle \) where:

- \( S' = (S \times [l_{\max} - 1]_{0} \times [W \cdot (l_{\max} - 1)]_{0}) \cup (S \times \{\text{trap}\}) \cup \{\text{trap}\}; \)
- \( E' \subseteq S' \times S'; \)
- \( w'(e = (s, x), (s', y)) = \bar{w}(s, s') \) if \( e \in E' \) and \((s, x), (s', y) \in S'; \)
- \( s'_{\text{init}} = (s_{\text{init}}, 0, 0); \)
- \( \mathbb{P}'((s, d, w), (s', \text{trap})) = p \) if \( \mathbb{P}(e = s \to s') = p, d = l_{\max} - 1 \) and \( w - \bar{w}(e) > 0; \)
- \( \mathbb{P}'((s, d, w), (s', d + 1, w - \bar{w}(e))) = p \) if \( \mathbb{P}(e = s \to s') = p, d < l_{\max} - 1 \) and \( w - \bar{w}(e) > 0; \)
- \( \mathbb{P}'((s, d, w), (s', 0, 0)) = p \) if \( \mathbb{P}(e = s \to s') = p \) and \( w - \bar{w}(e) \leq 0; \)
- \( \mathbb{P}'(\text{trap}, \text{trap}) = 1; \)
- \( \mathbb{P}'(\text{trap}, \text{trap}) = 1. \)

If the probability \( \mathbb{P}'(t, t') \) was not defined above for two states \( t, t' \in S' \), then \( \mathbb{P}'(t, t') = 0. \)

The idea of this construction is that if a path \( \pi \in \text{Paths}^{\mathcal{M}_{\max}^{\lambda}} \) visits a state of the form \((s, d, w)\), then it corresponds to a path \( \bar{\pi} \in \text{Paths}^{\mathcal{M}} \) that visits state \( s \). Moreover, the last window that is not closed (for threshold \( \lambda \)) when \( \bar{\pi} \) reaches the state \( s \) starts \( d < l_{\max} \) states before, and \( w \) is the deficit amount, that is, an additional weight of value \( w \) needs to be "seen" on the edges in order to close the window. If \( \pi \) is in state \text{trap}, then there is a window that could not be closed in \( \bar{\pi} \) within \( l_{\max} \) steps. In that case \( f_{\text{DirFixWMP}}^{\mathcal{M}_{\max}^{\lambda}}(\bar{\pi}) < \lambda \). The states of the form \((s, \text{trap})\) for \( s \in S \) are only here for the sake of the proof of the following theorem. When \( \mathcal{M}_{\max}^{\lambda} \) is constructed in Algorithm \([5]\) these states are not taken into account, and the edges arriving in them are directly forwarded to the state \text{trap}. The interest of these states is to avoid the possibility of having two edges coming from the same state arriving in the state \text{trap}.

The interest of this new Markov chain can be seen through the following theorem:

**Theorem 9.** Let \( \mathcal{M} \) be a Markov chain and \( \lambda \in \mathbb{Q} \) be a threshold. Then:

\[
\text{Pr}(f_{\text{DirFixWMP}}^{\mathcal{M}_{\max}^{\lambda}})^{-1}(\mathcal{M}, s_{\text{init}}, [\lambda, \infty[)) = \text{Pr}(\pi \in \mathcal{M}_{\max}^{\lambda} | \pi \models \neg \diamond \{\text{trap}\})
\]

An illustration of this construction can be seen in Figure \([4]\). In that example, the probability of eventually reaching the state \text{trap} is equal to 0.5 (that is the probability going to the state \( s_1 \) from the initial state). Therefore, the probability that the window mean-payoff is above 1.5 for a infinite path chosen randomly in the original Markov chain is also equal to 0.5.

In order to prove this result, we first consider the inductive property of windows (see \([7]\)).

**Inductive property of windows.** Let \( \rho = s_0 \ldots s_{|\rho|} \in \text{FPaths}^{\mathcal{M}}. \) Assume that there are \( j \leq j' < |\rho| \) such that the window opened at \( s_j \) is still open at \( s_{j'} \) and it is closed at \( s_{j'+1} \) (with respect to \( \lambda \)). Then, any window opened between \( s_j \) to \( s_{j'} \) (included) are closed in \( s_{j'+1} \) (with respect to \( \lambda \)).
Figure 4: The Markov chain $\mathcal{M}_{l_{\text{max}}}^\lambda$ obtained from the weighted Markov chain of Figure 1 with $l_{\text{max}} = 2$ and $\lambda = 1.5$. The weights have been modified ($w' = 2 \cdot w - 3$) and appear on each edge. If a state has one outgoing edge, the probability of taking it is 1, if it has two outgoing edges, the probability of taking each one of them is 0.5.
We can now see and prove the lemma that will be used in the proof of the theorem. We first denote by

\[
FP_{<\lambda}^M(s_{init}) = \{ \pi \in \text{Paths}_M^S(s_{init}) \mid |\pi| \geq l_{max} \land
\forall i < |\pi| - l_{max}, \text{WMP}(\pi(i \ldots (i + l_{max}))) \geq \lambda \land
\text{WMP}(\pi((|\pi| - l_{max}) \ldots (|\pi|)) < \lambda \}
\]

the set of finite paths in \( M \) where every window that was opened more than \( l_{max} \) states before the last state of the path is closed, with respect to \( \lambda \), and where the window that started \( l_{max} \) states before the last state could not be closed.

We also denote by

\[
FP_{\text{trap}}^{M_{l_{max}}}^{(s_{init})} = \{ \pi = s_0 \ldots s_{|\pi|} \in \text{Paths}_{M_{l_{max}}}^S(s_{init}) \mid \exists s \in S, s_{|\pi|} = (s, \text{trap}) \}
\]

the set of finite paths of \( M_{l_{max}}^\lambda \) that just entered a state \( (s, \text{trap}) \) for some \( s \in S \). We claim that:

**Lemma 10.** There exists a bijection \( h : FP_{<\lambda}^M(s_{init}) \rightarrow FP_{\text{trap}}^{M_{l_{max}}}^{(s_{init})} \) such that:

\[
\forall \rho \in FP_{<\lambda}^M(s_{init}), \n \exists \rho' = h(\rho)
\]

Proof. Let \( h : FP_{<\lambda}^M(s_{init}) \rightarrow FP_{\text{trap}}^{M_{l_{max}}}^{(s_{init})} \) be such that for \( \rho = s_0s_1 \ldots s_{|\rho|} \in FP_{<\lambda}^M(s_{init}) \), where \( s_0 = s_{init} \), we have \( h(\rho) = t_0t_1 \ldots t_{|\rho|} \in FP_{\text{trap}}^{M_{l_{max}}}^{(s_{init})} \) with

- \( t_0 = s'_{init} = (s_{init}, 0, 0) \in S' \)
- \( \forall i \in [|\rho| - 1] \)

\[
t_i = \begin{cases} (s_i, d + 1, v - w'(s_{i-1}, s_i)) & \text{if } v - w'(s_{i-1}, s_i) > 0 \\ (s_i, 0, 0) & \text{otherwise} \end{cases}
\]

where \( t_{i-1} = (s_{i-1}, d, v) \).

- \( t_{|\rho|} = (s_{|\rho|}, \text{trap}) \)

We first prove that \( h(\rho) \) is well defined. More specifically, we prove that, for any \( \rho \in FP_{<\lambda}^M(s_{init}) \), \( h(\rho) \in FP_{\text{trap}}^{M_{l_{max}}}^{(s_{init})} \).

Let \( \rho = (s_{init} = s_0) \ldots s_{|\rho|} \in FP_{<\lambda}^M(s_{init}) \). We prove the following inductive property, defined for \( l \in [|\rho| - 1] \):

\[
H(l) : t_0 \ldots t_l \in \text{Paths}_{M_{l_{max}}}^S(s_{init}), \text{ and } t_l = (s_l, d, v) \text{ where }
\]

(a) \( d \leq l_{max} - 1 \) is equal to the size of the largest window that is still opened in \( s_l \), that is, among all windows that are still opened in \( s_l \), the window that was opened the earliest (if no window is still opened, \( d = 0 \));

(b) \( v = -\sum_{k=1}^{d} w'(s_{l-k}, s_{l-k+1}) \geq 0 \). We note that if \( d = 0 \) then \( v = 0 \).
H(0) holds since \( t_0 = s_{\text{start}} = (s_{\text{init}}, 0, 0) \in \text{FPaths}^{\mathcal{M}_{l_{\text{max}}}}(s_{\text{start}}) \). Let us now assume that \( H(k-1) \) hold for \( k \leq l \in \{|\rho| - 1\} \). Let \( t_{l-1} = (s_{l-1}, d, v) \).

We first prove that \( t_0 \ldots t_{l} \in \text{FPaths}^{\mathcal{M}_{l_{\text{max}}}}(s_{\text{start}}) \). We assume towards a contradiction that \( d = l_{\text{max}} - 1 \) and \( v - w'(s_{l-1}, s_i) > 0 \). Because \( d = l_{\text{max}} - 1 \), this means that the window starting in \( s_{l-1} = s_{l_{\text{max}} - 1} \) is still open in \( s_{l-1} \). Moreover, \( v - w'(s_{l-1}, s_i) = - \sum_{k=1}^{l_{\text{max}}} w'(s_{l_{\text{max}} - 1 + k}, s_{l_{\text{max}} + k}) > 0 \). That is, \( \sum_{k=1}^{l_{\text{max}}} w(s_{l_{\text{max}} - 1 + k}, s_{l_{\text{max}} + k}) < l_{\text{max}} \cdot \frac{\alpha}{\rho} = l_{\text{max}} \cdot \lambda \) by definition of \( w' \). Therefore, the window opened in \( s_{l_{\text{max}} - 1} \) is still not closed in \( s_i \). Hence the contradiction since \( \rho \in \text{FP}_{\leq \lambda}(s_{\text{init}}) \), which implies that any window that was opened more than \( l_{\text{max}} \) states before the last state of \( \rho \) is closed \( (s_{l_{\text{max}} - 1} \) occurs more than \( l_{\text{max}} \) states before the last state of \( \rho \) since \( l \leq |\rho| - 1 \). We can conclude that \( d < l_{\text{max}} - 1 \lor v - w'(s_{l-1}, s_i) \leq 0 \) holds, and therefore \( \mathbb{P}(t_{l_{\text{max}} - 1} \mid t_{l_{\text{max}}}) = \mathbb{P}(s_{l_{\text{max}}}, s_i) > 0 \). In fact, \( t_0 \ldots t_{l} \in \text{FPaths}^{\mathcal{M}_{l_{\text{max}}}}(s_{\text{start}}) \).

Let us now consider conditions (a) and (b). If \( v - w'(s_{l-1}, s_i) > 0 \), it is obvious that (a) and (b) hold for \( t_{l} \). Let us now assume that \( v - w'(s_{l-1}, s_i) \leq 0 \) (which implies that the window opened in \( s_{l_{\text{max}} - 1} \) is closed in \( s_{l_{\text{max}} - 1} \)). In that case, \( t_1 = (s_i, 0, 0) \). We prove that any window opened before \( s_{l_{\text{max}} - 1} \) is closed. Because the window opened in \( s_{l_{\text{max}} - 1} \) is the earliest window to be still opened open at \( s_{l_{\text{max}} - 1} \), this implies that any window starting before \( s_{l_{\text{max}} - 1} \) is closed. Moreover, the inductive property of windows gives us that any window opened between \( s_{l_{\text{max}} - 1} \) and \( s_{l_{\text{max}} - 1} \) (included) is closed. Therefore any window opened before \( s_{l_{\text{max}} - 1} \) is closed. It follows that conditions (a) and (b) also hold in that case.

In fact, \( H(l) \) holds for all \( l \in \{|\rho| - 1\} \).

We now have to prove that \( \mathbb{P}(t_{l_{\text{max}} - 1} \mid t_l) = \mathbb{P}(s_{l_{\text{max}}}, s_l) > 0 \). That is, if we denote \( t_{l_{\text{max}} - 1} = (s_l, d, v) \), we have to prove that:

\[
d = l_{\text{max}} - 1 \land v - w'(s_{l_{\text{max}} - 1}, s_l) > 0
\]

Because \( \rho \in \text{FP}_{\leq \lambda}(s_{\text{init}}) \), the window opened in \( s_{l_{\text{max}} - 1} \) is not closed in \( s_l \). Therefore, according to \( H(l) \) for \( l \in \{|\rho| - 1\} \), the second component of the states \( t_l \) was never reset to 0 after \( t_{l_{\text{max}} - 1} \). In fact, \( d = l_{\text{max}} - 1 \). Moreover, according to \( H(|\rho| - 1) \), \( v - w'(s_{l_{\text{max}} - 1}, s_l) = - \sum_{k=0}^{l_{\text{max}}} w(s_{l_{\text{max}} - k}, s_{l_{\text{max}} - k}) > 0 \). We can conclude that \( h(\rho) = t_0 t_1 \ldots t_l \in \text{FP}_{\text{trap}}^{\mathcal{M}_{l_{\text{max}}}}(s_{\text{init}}) \).

\[\frac{\Pi \in \mathcal{M}_{l_{\text{max}}}}{\text{a bijection is straightforward from its definition, and the definition of } \mathcal{M}_{l_{\text{max}}}}\]

The fact that \( \forall \rho \in \text{FP}_{\leq \lambda}(s_{\text{init}}) \), \( \mathbb{P}(\rho) = \mathbb{P}(h(\rho)) \) comes from the definition of \( \mathcal{M}_{l_{\text{max}}} \), where the probabilities are the same as in the original Markov chain \( \mathcal{M} \).

We can now proceed to the proof of Theorem \( \Box \).

**Proof.** We have that:

\[
\text{Pr}(\{ \pi \in \mathcal{M}_{l_{\text{max}}} \mid \pi \models \Diamond \{\text{trap}\}\}) = \text{Pr}(\{ \pi \in \mathcal{M}_{l_{\text{max}}} \mid \pi \models \Diamond \{s, \text{trap}\} \mid s \in S\})
\]

by definition of the Markov chain \( \mathcal{M}_{l_{\text{max}}} \). Moreover, by definition of the sets \( \text{FP}_{\leq \lambda}(s_{\text{init}}) \) and
It follows that:

\[
\left(f^{-1}_{\text{DirFixWMP}}(\mathcal{M}, s_{\text{init}}) - \infty, \lambda\right] = \bigcup_{\pi \in \mathcal{F P}_{\lambda}^{\mathcal{M}}(s_{\text{init}})} \text{Cyl}(\pi)
\]

\[
\{\pi \in \mathcal{M}_{\text{max}}^{\lambda} \mid \pi = \Diamond\{(s, \text{trap}) \mid s \in S\}\} = \bigcup_{\pi \in \mathcal{F P}_{\text{trap}}^{\mathcal{M}}(s'_{\text{init}})} \text{Cyl}(\pi)
\]

Let us denote by \( h \) the function defined in Lemma 10. Then, we have:

\[
\Pr\left((f^{-1}_{\text{DirFixWMP}}(\mathcal{M}, s_{\text{init}}) - \infty, \lambda)\right] = \sum_{\pi \in \mathcal{F P}_{\lambda}^{\mathcal{M}}(s_{\text{init}})} \Pr(\text{Cyl}(\pi))
\]

\[
= \sum_{\pi \in \mathcal{F P}_{\lambda}^{\mathcal{M}}(s_{\text{init}})} \Pr(\text{Cyl}(h(\pi)))
\]

\[
= \sum_{\pi \in \mathcal{F P}_{\lambda}^{\mathcal{M}}(s_{\text{init}})} \Pr\left(\bigcup_{\pi \in \mathcal{F P}_{\lambda}^{\mathcal{M}}(s_{\text{init}})} \text{Cyl}(h(\pi))\right)
\]

\[
= \Pr\left(\bigcup_{\pi \in \mathcal{F P}_{\lambda}^{\mathcal{M}}(s_{\text{init}})} \text{Cyl}(\pi)\right) = \Pr\left(\{\pi \in \mathcal{M}_{\text{max}}^{\lambda} \mid \pi = \Diamond\{(s, \text{trap}) \mid s \in S\}\}\right)
\]

It follows that:

\[
\Pr((f^{-1}_{\text{DirFixWMP}}(\mathcal{M}, s_{\text{init}}, [\lambda, \infty])) = 1 - \Pr((f^{-1}_{\text{DirFixWMP}}(\mathcal{M}, s_{\text{init}}) - \infty, \lambda))
\]

\[
= 1 - \Pr(\{\pi \in \mathcal{M}_{\text{max}}^{\lambda} \mid \pi = \Diamond\{(s, \text{trap}) \mid s \in S\}\})
\]

\[
= 1 - \Pr(\{\pi \in \mathcal{M}_{\text{max}}^{\lambda} \mid \pi = \Diamond\{\text{trap}\}\})
\]

\[
= \Pr(\{\pi \in \mathcal{M}_{\text{max}}^{\lambda} \mid \pi = \neg\Diamond\{\text{trap}\}\})
\]

This theorem enables us to compute the probability that the value of a path is above a given threshold. However, to design an algorithm that computes the probabilities of various values of \( \lambda \) (and then to compute the expected value), we need an additional lemma that specifies the set of possible values of \( \lambda \). We observe that for \( \pi \in \text{Paths}^{M}(s_{\text{init}}), \{\text{WMP}(\pi(i \ldots (i + l_{\text{max}})) \mid i \in \mathbb{N}\} \subseteq \{\text{WMP}(\pi(0 \ldots l_{\text{max}})) \mid \pi \in \text{Paths}^{M}_{l_{\text{max}}}) \subseteq \{\frac{E}{q} \mid q \in [l_{\text{max}}], p \in [W \cdot q]_0\} \) which is finite. Hence, the following lemma:

**Lemma 11.** Let the set \( \text{PossibleVal} = \{\lambda \in \mathbb{R} \mid \Pr((f^{-1}_{\text{DirFixWMP}}(\mathcal{M}, s_{\text{init}}, \lambda) > 0\}. \)

Then,

\[
\text{PossibleVal} \subseteq \left\{\frac{p}{q} \mid q \in [l_{\text{max}}], p \in [W \cdot q]_0\}
\]

where \( W = \max_{e \in E} w(e) \).

By definition of the set \( \text{PossibleVal} \), we have:

\[
\mathbb{E}_{s_{\text{init}}}^{M}(f^{-1}_{\text{DirFixWMP}}) = \sum_{\lambda \in \text{PossibleVal}} \lambda \cdot \Pr((f^{-1}_{\text{DirFixWMP}}(\mathcal{M}, s_{\text{init}}, \lambda))
\]

(9)

This lemma ensures that the only values of interest are of the form \( \frac{p}{q} \), where \( p \in [W \cdot l_{\text{max}}]_0 \) and \( q \in [l_{\text{max}}] \). That is what is used in Algorithm 5.

Algorithm 5 computes the expected value of \( f^{-1}_{\text{DirFixWMP}} \) in a Markov chain. At the end of the two for loops (line 9), for all \( \lambda \in \text{PossibleVal} \), we have \( \text{Val}(\lambda) = \Pr((f^{-1}_{\text{DirFixWMP}}(\mathcal{M}, s_{\text{init}}, [\lambda, \infty])) \).
Then, the value $\Pr((f_{\text{DirFixWMP}}^{l_{\max}})^{-1}(M, s_{\text{init}}, \lambda))$ is computed (line 12) using the fact that if we have $\text{PossibleVal} = \{\lambda_1, \lambda_2, \ldots, \lambda_n\}$ with $\lambda_1 > \lambda_2 \ldots > \lambda_n$ then, for all $i \in \{2, \ldots, n\}$:

$$\Pr((f_{\text{DirFixWMP}}^{l_{\max}})^{-1}(M, s_{\text{init}}, \lambda_i)) = \Pr((f_{\text{DirFixWMP}}^{l_{\max}})^{-1}(M, s_{\text{init}}, [\lambda, \infty[))$$

and

$$\Pr((f_{\text{DirFixWMP}}^{l_{\max}})^{-1}(M, s_{\text{init}}, \lambda_1)) = \Pr((f_{\text{DirFixWMP}}^{l_{\max}})^{-1}(M, s_{\text{init}}, [\lambda_1, \infty[))$$

Finally, the expected value of $f_{\text{DirFixWMP}}^{l_{\max}}$ is computed by using Formula 9.

**Algorithm 5** DirFixWMP$(M, s_{\text{init}}, l_{\max})$

**Require:** $M = (S, E, s_{\text{init}}, P, w)$ is a weighted Markov Chain, $s_{\text{init}} \in S$ and $l_{\max} \in \mathbb{N}_0$

**Ensure:** $E$ is equal to $E_{s_{\text{init}}}^{M}(f_{\text{DirFixWMP}}^{l_{\max}})$

1. $W = \max_{e \in E} |w(e)|$
2. $Val := \text{AssocTable}(\text{float}, \text{float})$ : first elements are sorted in descending order
3. for $q \in [l_{\max}]$ do
4.  for $p \in [W \cdot q]|0$ do
5.  \hspace{1cm} $\lambda = \frac{p}{q}$
6.  \hspace{1cm} $M_{\lambda}^{l_{\max}} := (S', P', s_{\text{start}})$
7.  \hspace{1cm} $\text{Prob}_{\geq \lambda} := \Pr_{s_{\text{start}}}(\diamond \{\text{trap}\})$
8.  \hspace{1cm} $Val(\lambda) := \text{Prob}_{\geq \lambda}$
9. \hspace{1.5cm} $E := 0$
10. \hspace{1.5cm} $\text{Prob}_{> \lambda} = 0$
11. for $\lambda \in Val$ do
12. \hspace{1cm} $\text{Prob}_{= \lambda} := Val(\lambda) - \text{Prob}_{> \lambda}$
13. \hspace{1cm} $E := E + \lambda \cdot \text{Prob}_{= \lambda}$
14. \hspace{1cm} $\text{Prob}_{\leq \lambda} := Val(\lambda)$

return $E$

Let us now consider the complexity of Algorithm 5. We enter $(W + 1) + 2 \cdot (W + 1) + \ldots + l_{\max} \cdot (W + 1) = O(W \cdot l_{\max}^2)$ times the lines 5-8. For $\lambda = \frac{p}{q}$, the size of the Markov chain $M_{\lambda}^{l_{\max}}$ that is constructed line 6 is of size $O(|S| \cdot l_{\max} \cdot (W \cdot q - p) \cdot l_{\max}) = O(|S| \cdot l_{\max} \cdot W \cdot l_{\max} \cdot l_{\max})$. The factor $W \cdot q - p$ comes from the fact that we consider a new weight function $w' = q \cdot w - p$. Then, computing the probability to reach the state $\text{trap}$ is in $O(\text{Mat}(|S| \cdot l_{\max}^3 \cdot W))$. Therefore, the complexity of the algorithm is in $O(W \cdot l_{\max}^2 \cdot \text{Mat}(|S| \cdot l_{\max}^3 \cdot W))$.

We also have a different algorithm to compute $E_{s_{\text{init}}}^{M}(f_{\text{DirFixWMP}}^{l_{\max}})$ based on another construction of Markov chains. That new algorithm is exponential in the window size $l_{\max}$ but is logarithmic in $W$, the maximal weight appearing in the Markov chain. It may be faster than the previous algorithm in the cases where the weights in the Markov chain are very high. The details are given in Appendix C.

### 4 Window Mean-Payoff in Weighted Markov Decision processes

In this section, given an MDP, we study the problems of finding the optimal expected values of the functions defined in Section 2.6 over all the strategies available in the MDP.

We start with the prefix independent version of the fixed window objective function and show an algorithm that computes the expected value of the objective function in time that is polynomial in the the size of the MDP and the window size. Further, an optimal strategy
Next we show that the bounded window mean-payoff problem can be solved in time \(NP \cap coNP\) and a memoryless optimal strategy exists. Finally, we study the direct fixed window objective function and find its expected value using an algorithm that runs in time that is exponential in the window length. An optimal strategy also requires a finite memory that is bounded above by \(W^{l_{max}} \cdot l_{max}^2\), where \(W\) is its maximum weight and \(l_{max}\) is the window length. We show a PSPACE-hardness for the problem even when \(l_{max}\) is given in unary thus excluding the possibility of having an algorithm that is pseudopolynomial unless \(P=\text{PSPACE}\). For the prefix independent version of the fixed window objective function, we show that the problem of computing an expected value is at least as hard as computing the value in a two-player window mean-payoff game with the fixed window objective. For the optimal expected value of the bounded window mean-payoff objective function, we show that the problem of computing the value is at least as hard as solving the value problem in traditional mean-payoff games.

In the following, we will refer to the optimal expected value as the expected value in the MDP for the corresponding function.

### 4.1 Fixed Window Mean-Payoff

Consider an MDP \(\Gamma = \langle S, E, \text{Act}, s_{init}, w, \mathbb{P} \rangle\). Then, let

\[
E_{s_{init}}^\Gamma (f_{FixWMP}^{l_{max}}) = \sup_{\sigma \in \text{strat}(\Gamma)} E_{s_{init}}^{\Gamma[\sigma]} (f_{FixWMP}^{l_{max}})
\]

We first design an algorithm that solves this problem in time \(O(\log(W \cdot l_{max}^2) \cdot (|S| \cdot |\text{Act}|) \cdot (|S| + |E|) \cdot l_{max} \cdot \log(W))\) where \(W = \max w(e)\). Then, we show that the problem of deciding the two-player game for the direct fixed window mean objective can be reduced to the expected fixed window mean-payoff in an MDP in polynomial time.

#### 4.1.1 Algorithm

Since the fixed window mean-payoff is prefix independent, we first look at the maximal end components (recall that, for every strategy \(\sigma\), each path will almost surely end up in an MEC, by Proposition 2). Consider an MEC \(M = (T, A) \in MEC(\Gamma)\) of \(\Gamma\) and a state \(s \in T\). Since the subgraph induced by \(M\) is strongly connected (since it is an MEC), for every pair of states \(s, s' \in T\), there exists a reaching strategy \(\sigma_{s,s'} \in \text{strat}(\Gamma)\) such that every path starting from \(s\) reaches \(s'\) almost surely, in the Markov chain \(\Gamma[\sigma_{s,s'}]\). Therefore, if we denote by \(v \in T\) a state that maximizes the expected value \(E_s^\Gamma (f_{FixWMP}^{l_{max}})\) among all states \(s \in T\), that is \(E_s^\Gamma (f_{FixWMP}^{l_{max}}) = \max_{s \in T} (E_s^\Gamma (f_{FixWMP}^{l_{max}}))\), then \(v\) can be reached almost surely from every state \(s \in T\). Since \(f_{FixWMP}^{l_{max}}\) is prefix independent, we have:

\[
\forall s \in T, \ E_s^\Gamma (f_{FixWMP}^{l_{max}}) = E_v^\Gamma (f_{FixWMP}^{l_{max}})
\]

This is true in every MEC of \(\Gamma\). For each \(M \in MEC(\Gamma)\), we denote by \(\lambda_{M}^{l_{max}}\) the (optimal) expected value of \(f_{FixWMP}^{l_{max}}\) from every state of \(M\), computed among the paths that stay in \(M\), that is \(\lambda_{M}^{l_{max}} = E_M^\Gamma (f_{FixWMP}^{l_{max}})\). In fact, the value \(\lambda_{M}^{l_{max}}\) can be expressed using a two-player game. Formally, we have the following theorem:

**Theorem 12.** Let \(M = (T, A)\) be an MEC of \(\Gamma\). Then we have:

\[
\lambda_{M}^{l_{max}} = \max_{s \in T} 2 \cdot \sup_{\sigma_1 \in \text{strat}_1(G_M)} \inf_{\sigma_2 \in \text{strat}_2(G_M)} V_{s_{\text{DirFixWMP}}}^{l_{max}}(\langle G_M, \sigma_1, \sigma_2 \rangle) \noted y(s)
\]
Before proving this theorem, we make a few observations. The value \( g(s) \) denotes the outcome (multiplied by 2) of the two-player game resulting from \( M \) where the function of interest is the direct fixed window mean-payoff function for a window length \( 2l_{\text{max}} \). From [7], we know that deterministic finite memory strategies are enough to solve the two-player game for the direct fixed window mean-payoff. Moreover, the memory needed to solve that game is bounded above by \( |S| \cdot l \) with \( l = 2 \cdot l_{\text{max}} \) and \( S \) denotes the total number of states in the game.

When we construct a two-player game from an MDP, an edge of weight \( w \) is split into two edges (hence, doubling the number of edges seen on a path) of weight 0 and \( w \) (which divides the mean-payoff of a path by 2). Therefore, if the fixed window mean-payoff of a path \( \pi \) in the MDP equals \( 2 \cdot \lambda \) for a window length \( l_{\text{max}} \), then the fixed window mean-payoff over the corresponding path in the two-player game equals \( \lambda \), for a window length of \( 2 \cdot l_{\text{max}} \).

**Proof.** We proceed in two steps:

1. First we show that \( \lambda_M^l \geq \max_{s \in T} g(s) \). Let \( \sigma_1, \sigma_2 \) be two optimal strategies for Player 1 and Player 2 respectively from a state \( v \in T \) that ensures \( g(v) = \max_{s \in T} g(s) \) (\( v \) is called an optimal state). Let \( \pi \in \text{Paths}_v^M \). Let \( \sigma_2^g \in \text{strat}_2(G_M) \) be a (deterministic) strategy for Player 2 such that the outcome of the game \( G_M \) played with strategies \( \sigma_1 \) and \( \sigma_2^g \) follows exactly the sequence of states visited by \( \pi \) in \( M^{[\sigma_1]} \). Formally, \( \sigma_2^g \) ensures that \( \forall (G_M, v, \sigma_1, \sigma_2^g) = p_{\sigma_1}(\pi) \). By definition of \( \sigma_2 \), we have \( f(v, \sigma_1, \sigma_2^g) = 2 \cdot f_{\text{DirF}\times\text{WMP}}(\sigma_1(G_M, v, \sigma_1, \sigma_2^g)) = 2 \cdot f_{\text{DirF}\times\text{WMP}}^l(p_{\sigma_1}(\pi)) \leq f_{\text{FixWMP}}^l(\pi) \).

2. Now, we show that \( \lambda_M^l \leq \max_{s \in T} g(s) \). Let \( \sigma_1 \in \text{strat}(M) \) be a strategy in the MDP and let \( \pi' \in \text{Paths}_v^M \) for some \( t \in T \). For every state \( s \) occurring in \( \pi' \), there exists a strategy \( \sigma_s \in \text{strat}_2(G_M) \) such that \( 2 \cdot f_{\text{DirF}\times\text{WMP}}^l(\pi(G_M, s, \sigma_1, \sigma_s)) \leq g(s) \) (it may be lower than \( g(s) \) since \( \sigma_1 \) may not be an optimal strategy for Player 1). Therefore, in the MDP \( M \), there exists a finite sequence of states, that starts in \( s \), and that contains a sequence of \( l_{\text{max}} \) states whose window mean-payoff is lower than or equal to \( g(s) \), \( \forall s \in T \). Thus, the probability of seeing such a sequence of states from every occurrence of \( s \) is strictly positive. This is true for every state \( s \) occurring in \( \pi' \). Hence, infinitely often, there is a non-zero probability of seeing a sequence of states for which the window mean-payoff is lower than or equal to \( g(s) \). Thus each path almost surely has a fixed window mean-payoff of at most \( \max_{s \in T} g(s) \).

\[ \Pr((f_{\text{FixWMP}}^l)^{-1}(M^{[\sigma_1]}, t, [0, \max_{s \in T} g(s)])) = 1, \]

which implies that \( \mathbb{E}_v^M(f_{\text{FixWMP}}^l) \leq \max_{s \in T} g(s) \). This is true for every strategy \( \sigma_1 \in \text{strat}(M) \).

---

\[ ^3 \text{Since there are finitely many strategies with a fixed memory size, the supremum and infimum become maximum and minimum respectively. Also since the two-player game with direct fixed window objective is determined, the sup and the inf can be interchanged.} \]
is, \( \forall \sigma \in \text{strat}(M), \mathbb{E}_{\pi_{I}}^{M[\sigma]}(f_{\text{FixWMP}}^{\max}) \leq \max_{s \in T} g(s) \). Therefore, \( \mathbb{E}_{I}^{M}(f_{\text{FixWMP}}^{\max}) \leq \max_{s \in T} g(s) \).

Thus, \( \lambda_{M}^{\max} \leq \max_{s \in T} g(s) \).

The theorem follows. This (especially the first paragraph) also proves that an optimal strategy for Player 1 in the two-player game \( G_{M} \) is also an optimal strategy in the MDP \( M \). Therefore, there exists an optimal strategy for which the expected value of the fixed window mean-payoff is equal to \( \lambda_{M}^{\max} \) and whose memory size bounded above by \( |T| \cdot l_{\max} \). \( \square \)

**Remark 13.** This proof also shows that the theorem holds even if the objective in the two-player game is the fixed window mean-payoff objective. This comes from the fact that the inequality \( g(v) \leq f_{\text{FixWMP}}^{\max}(\pi) \) (in the first paragraph) also holds if we consider that new objective. We stated Theorem 12 with the direct fixed window mean-payoff objective because the algorithm presented in [7] to solve the two-player for that objective is faster than the algorithm presented in [7] to solve the fixed window mean-payoff objective.

We now compute the expected value in the whole MDP. We denote by \( \Gamma^{\text{MEC}} = (S, E, \text{Act}, s_{\text{init}}, w, \mathbb{P}) \) the MDP where for every \( M \in \text{MEC}(\Gamma) \), for every edge \( e = (s, a, s') \in E \) that appears in \( M \), the weight on the edge \( e \) is replaced by \( \lambda_{M}^{\max} \). Formally, for all edges \( e = (s, a, s') \in E \):

- \( w'(s, a, s') = \lambda_{M}^{\max} \) if \( M = (T, A) \in \text{MEC}(\Gamma) \) and \( a \in A \);
- \( w'(s, a, s') = w(s, a, s') \) otherwise.

For every \( M \in \text{MEC}(\Gamma) \), we denote by \( M_{\Lambda} \in \text{MEC}(\Gamma^{\text{MEC}}) \) the corresponding maximal end component in \( \Gamma^{\text{MEC}} \) in which all the weights have been replaced by \( \lambda_{M}^{\max} \).

For a Markov chain \( \mathcal{M} = (S, E, s_{\text{init}}, w, \mathbb{P}) \) and for \( \pi = s_{0} \ldots \in \text{Paths}^{\mathcal{M}} \), we define the function \( f_{\text{Mean}} : \text{Paths}^{\mathcal{M}} \rightarrow \mathbb{R} \) such that \( f_{\text{Mean}}(\pi) = \lim_{n \to \infty} \frac{1}{n} \sum_{k=0}^{n-1} w(s_{k}, s_{k+1}) \). This corresponds to the mean-payoff of the path \( \pi \). Now, we have the following theorem, which states that the expected value of the fixed window mean-payoff in \( \Gamma \), that is \( \mathbb{E}_{s_{\text{init}}}^{\Gamma}(f_{\text{FixWMP}}^{\max}) \), is equal to the expected value of the mean-payoff in \( \Gamma^{\text{MEC}} \), that is \( \mathbb{E}_{s_{\text{init}}}^{\Gamma^{\text{MEC}}}(f_{\text{Mean}}) \):

**Theorem 14.** For any Markov decision process \( \Gamma \), and any initial state \( s_{\text{init}} \), we have:

\[
\mathbb{E}_{s_{\text{init}}}^{\Gamma}(f_{\text{FixWMP}}^{\max}) = \mathbb{E}_{s_{\text{init}}}^{\Gamma^{\text{MEC}}}(f_{\text{Mean}})
\]

**Proof.** Consider an MDP \( \Gamma \). By definition, \( \Gamma \) and \( \Gamma^{\text{MEC}} \) only differ in the weights inside the MECs. Therefore, \( \text{strat}(\Gamma) = \text{strat}(\Gamma^{\text{MEC}}) \), and for each \( \sigma \in \text{strat}(\Gamma) \), we have \( \text{Paths}^{\Gamma^{\text{MEC}}[\sigma]}(s_{\text{init}}) = \text{Paths}^{\Gamma^{\text{MEC}}}(s_{\text{init}}) \).

We also proceed in two steps:

1. We first prove that \( \mathbb{E}_{s_{\text{init}}}^{\Gamma}(f_{\text{FixWMP}}^{\max}) \leq \mathbb{E}_{s_{\text{init}}}^{\Gamma^{\text{MEC}}}(f_{\text{Mean}}) \). Let \( \sigma \in \text{strat}(\Gamma) \). Let \( M \in \text{MEC}(\Gamma) \) be a maximal end component. Then, we have:

   - Since \( \sigma \) is any arbitrary strategy, the expected value of \( f_{\text{FixWMP}}^{\max} \) in \( M[\sigma] \) ensures \( \mathbb{E}_{M[\sigma]}^{M}(f_{\text{FixWMP}}^{\max}) \leq \lambda_{M}^{\max} \), by definition of \( \lambda_{M}^{\max} \);  
   - the construction of \( M_{\Lambda}[\sigma] \) in \( \Gamma^{\text{MEC}} \) ensures that the expected value of \( f_{\text{Mean}} \) in \( M_{\Lambda}[\sigma] \) equals \( \lambda_{M_{\Lambda}}^{\max} \) that is, \( \mathbb{E}_{M_{\Lambda}[\sigma]}^{M_{\Lambda}}(f_{\text{Mean}}) = \lambda_{M_{\Lambda}}^{\max} \) since every weight appearing in \( M_{\Lambda}[\sigma] \) is equal to \( \lambda_{M_{\Lambda}}^{\max} \).

   Therefore, \( \mathbb{E}_{M[\sigma]}^{M}(f_{\text{FixWMP}}^{\max}) \leq \mathbb{E}_{M_{\Lambda}[\sigma]}^{M_{\Lambda}}(f_{\text{Mean}}) \);

2. We then prove that \( \mathbb{E}_{s_{\text{init}}}^{\Gamma}(f_{\text{FixWMP}}^{\max}) \leq \mathbb{E}_{s_{\text{init}}}^{\Gamma^{\text{MEC}}}(f_{\text{Mean}}) \). Let \( \sigma \in \text{strat}(\Gamma) \). Let \( M \in \text{MEC}(\Gamma) \) be a maximal end component. Then, we have:

   - Since \( \sigma \) is any arbitrary strategy, the expected value of \( f_{\text{FixWMP}}^{\max} \) in \( M[\sigma] \) ensures \( \mathbb{E}_{M[\sigma]}^{M}(f_{\text{FixWMP}}^{\max}) \leq \lambda_{M}^{\max} \), by definition of \( \lambda_{M}^{\max} \);  
   - the construction of \( M_{\Lambda}[\sigma] \) in \( \Gamma^{\text{MEC}} \) ensures that the expected value of \( f_{\text{Mean}} \) in \( M_{\Lambda}[\sigma] \) equals \( \lambda_{M_{\Lambda}}^{\max} \) that is, \( \mathbb{E}_{M_{\Lambda}[\sigma]}^{M_{\Lambda}}(f_{\text{Mean}}) = \lambda_{M_{\Lambda}}^{\max} \) since every weight appearing in \( M_{\Lambda}[\sigma] \) is equal to \( \lambda_{M_{\Lambda}}^{\max} \).

   Therefore, \( \mathbb{E}_{M[\sigma]}^{M}(f_{\text{FixWMP}}^{\max}) \leq \mathbb{E}_{M_{\Lambda}[\sigma]}^{M_{\Lambda}}(f_{\text{Mean}}) \);
– The probability of eventually staying forever in \( M \) is equal to the probability of eventually staying forever in \( M \). That is, \( Pr^\sigma(\Diamond W | M) = Pr^{\Gamma_{MEC} \strat(\sigma)}(\Diamond W_M) \).

Moreover, we know that every path (in \( \Gamma \) and in \( \Gamma_{MEC} \)) will end up, almost surely, in an MEC. Therefore:

\[
\mathbb{E}^{\Gamma_{\strat}}_{\text{init}}(f_{\text{FixWMP}}^{\max}) = \sum_{M \in \text{MEC}(\Gamma)} \mathbb{E}^{\Gamma_{\strat}}_{\text{init}}(f_{\text{FixWMP}}^{\max}) \cdot Pr^{\sigma}(\Diamond W | M) \cdot \mathbb{E}^{\sigma}_{\text{init}}(f_{\text{FixWMP}}^{\max}) \leq \mathbb{E}^{\Gamma_{MEC}}_{\text{init}}(f_{\text{Mean}}),
\]

This is true for all \( \sigma \in \strat(\Gamma) \), that is, for all \( \sigma \in \strat(\Gamma) \), \( \mathbb{E}^{\Gamma_{\strat}}_{\text{init}}(f_{\text{FixWMP}}^{\max}) \leq \mathbb{E}^{\Gamma_{MEC}}_{\text{init}}(f_{\text{Mean}}) \). Thus, \( \mathbb{E}^{\Gamma}_{\strat}(f_{\text{FixWMP}}^{\max}) \leq \mathbb{E}^{\Gamma_{MEC}}_{\text{init}}(f_{\text{Mean}}) \).

(2) We now prove that \( \mathbb{E}^{\Gamma}_{\strat}(f_{\text{FixWMP}}^{\max}) \geq \mathbb{E}^{\Gamma_{MEC}}_{\text{init}}(f_{\text{Mean}}) \). We know that, given an MDP \( \Gamma \), there exists an optimal memoryless strategy that maximizes the expected mean-payoff in \( \Gamma \) (see [9]). Thus, let us consider a memoryless strategy \( \sigma \in \strat(\Gamma) \) that maximizes the expected value of the mean-payoff in \( \Gamma_{MEC} \). We construct a strategy \( \sigma' \in \strat(\Gamma) \) such that \( \mathbb{E}^{\Gamma_{\strat}}_{\text{init}}(f_{\text{FixWMP}}^{\max}) = \mathbb{E}^{\Gamma_{MEC}}_{\text{init}}(f_{\text{Mean}}) \). Let us denote by \( M(\sigma) \) the set of MECs of \( \Gamma \) that contains at least one BSCC reachable from \( \text{init} \) of the Markov chain \( \Gamma_{\strat} \) (there is not necessarily a BSCC in every MEC, if an MEC \( M \) is not interesting to end up in, in terms of maximizing the expected mean-payoff. In such case \( \sigma \) leaves \( M \) and thus no BSCC will appear in \( M \)). Let \( \sigma' \in \strat(\Gamma) \) be a strategy that acts like \( \sigma \) outside the maximal end components in \( M(\sigma) \) and, inside an MEC \( M \in M(\sigma) \), acts to get an expected value equal to \( \lambda_{\text{max}}^{\text{fix}} \) while staying inside \( M \) (this is possible with a finite memory strategy, as we have seen in the proof of Theorem [12]). In this way, once inside an MEC \( M \in M(\sigma) \), the expected value of \( f_{\text{FixWMP}}^{\max} \) in \( M \) (that is \( \lambda_{\text{max}}^{\text{fix}} \), by definition of the strategy \( \sigma' \)) is the same as the expected value of \( f_{\text{Mean}} \) in \( \Gamma_{MEC} \) (also \( \lambda_{\text{max}}^{\text{fix}} \), since every weights appearing inside \( M \) are equal to \( \lambda_{\text{max}}^{\text{fix}} \)). Moreover, since the strategy \( \sigma' \) acts like \( \sigma \) outside the MECs \( M \in M(\sigma) \), the probability of reaching every MEC \( M \in M(\sigma) \) is the same in both \( \Gamma_{\strat} \) and in \( \Gamma_{MEC} \). Hence, \( \mathbb{E}^{\Gamma_{\strat}}_{\text{init}}(f_{\text{FixWMP}}^{\max}) = \mathbb{E}^{\Gamma_{MEC}}_{\text{init}}(f_{\text{Mean}}) \). By choice of \( \sigma \), we have \( \mathbb{E}^{\Gamma_{MEC}}_{\text{init}}(f_{\text{Mean}}) = \mathbb{E}^{\Gamma_{MEC}}_{\text{init}}(f_{\text{Mean}}) \). Therefore, \( \mathbb{E}^{\Gamma}_{\strat}(f_{\text{FixWMP}}^{\max}) = \sup_{\sigma' \in \strat(\Gamma)} \mathbb{E}^{\Gamma_{\strat}}_{\text{init}}(f_{\text{FixWMP}}^{\max}) \geq \mathbb{E}^{\Gamma_{MEC}}_{\text{init}}(f_{\text{Mean}}) \).

This concludes the proof. \( \square \)

**Corollary 14.1.** The memory size that is needed to construct an optimal strategy \( \sigma' \) that maximizes the expected value of \( f_{\text{FixWMP}}^{\max} \) in an MDP \( \Gamma = (S, E, \text{Act}, \text{init}, w, \mathbb{P}) \) is bounded above by \(|S| \cdot I_{\text{max}}\).

**Proof.** One can construct an optimal strategy \( \sigma' \) as follows:

- The first part of \( \sigma' \) in \( \Gamma \) follows the strategy \( \sigma \) in \( \Gamma_{MEC} \) that is used to maximize the expected value of the mean-payoff until it reaches some MEC \( M \) such that \( \sigma \) induces one or more BSCCs in \( M \). Recall that the corresponding MEC \( M \) also exists in \( \Gamma \);

- Then, once inside an MEC \( M = (T, A) \) in which it is possible to end up by following \( \sigma \) in \( \Gamma \), we apply a strategy so that an optimal state \( v \in T \) maximizing \( g(s) \) over all states \( s \in T \) is reached almost surely;

- Finally, we apply the optimal strategy that maximizes the outcome of the two-player window mean-payoff game from the optimal state \( v \).
The first two steps can be done with a memoryless strategy. The third one requires a memory size bounded above by \(|T| \cdot l_{\text{max}}|\) for every MEC \(M = (T, A)\) in which it is possible to end up by following \(\sigma\). Hence the memory required is bounded above by \(|S| \cdot l_{\text{max}}|\), where \(S\) is the set of states in \(\Gamma\).

We now examine the algorithms that computes these expected values. In [7], Algorithm \textit{DirectFWMP}(\(G, l_{\text{max}}\)) returns the set of winning states for the direct fixed window mean-payoff objective in the two-player game \(G = (S_1, S_2, s_{\text{init}}, E, w)\) with threshold 0. This is done in time \(O(|S_1 \cup S_2| \cdot |E| \cdot l_{\text{max}}| \cdot \log(W))\) where \(W = \max_{e \in E} w(e)\).

\begin{algorithm}
\caption{\textsc{FixWMP}_{\text{MEC}}(\(\Gamma, l_{\text{max}}\))}
\begin{algorithmic}
\Require \(M = (T, E, A, s_{\text{init}}, w, \mathcal{P})\) is a weighted Markov decision process that is a maximal end component, \(l_{\text{max}} \in \mathbb{N}_0\)
\Ensure \(\mu\) is equal to \(\mathcal{E}^M(f_{\text{FixWMP}}^{l_{\text{max}}})\)
\State \(G_\Gamma := (T_1, T_2, s_{\text{init}}, E, w')\) : the two-player game obtained from the MDP \(M\)
\State \(W = \max_{e \in E} w(e)\)
\State \(L_B := 0, U_B := W + 1\)
\State \(\epsilon := L_B + U_B\)
\While {\(\epsilon > \frac{1}{l_{\text{max}}}\)}
\State \(\lambda := \frac{L_B + U_B}{2}\)
\State \(\epsilon := \frac{\epsilon}{2}\)
\State \(T' := \text{DirectFWMP}(\langle T_1, T_2, s_{\text{init}}, E, w' - \lambda/2 \rangle, 2 \cdot l_{\text{max}})\)
\If {\(T' \neq \emptyset\)}
\State \(L_B := \lambda\)
\Else
\State \(U_B := \lambda\)
\EndIf
\For {\(l \in [l_{\text{max}}]\)}
\State \(l_B = l \times L_B; u_B = l \times U_B\)
\If {\(\lfloor l_B \rfloor + 1 = \lfloor u_B \rfloor\)}
\State \(\mu := \frac{u_B}{l}\)
\Break
\EndIf
\EndFor
\EndWhile
\State \Return \(\mu\)
\end{algorithmic}
\end{algorithm}

Algorithm [8] computes the expected value of \(f_{\text{FixWMP}}^{l_{\text{max}}}\) inside an MEC by solving the two-player window mean-payoff game, using the result of Theorem[12]. Additionally, a binary search very close to the one used in Algorithm[2] is used. This binary search looks for \(\lambda_M^{l_{\text{max}}}\) between the two bounds 0 and \(W\). Recall that we assume that \(\Gamma\) has non-negative integer weights. However, like mentioned before, the value of a path inside the MDP is twice as much as its value in the two-player game. That is why, in line 8 (where this algorithm differs from Algorithm[2]), the call to Algorithm \textit{DirectFWMP} is done with the weights reduced by \(\lambda/2\) instead of \(\lambda\).

Let us now discuss its complexity. Similar to Algorithm[2] we enter the while loop \(O(\log(W \cdot l_{\text{max}}^2))\) times. The most expensive operation is the call to algorithm \textit{DirectFWMP}, that runs in time \(O((|T_1| + |T_2|) \cdot |E| \cdot l_{\text{max}} \cdot \log(W))\) with \(|T_1| = |T|\) and \(|T_2| = |T| \cdot |A|\). Therefore, the complexity of Algorithm[8] is \(O(\log(W \cdot l_{\text{max}}^2) \cdot (|T| \cdot |A|) \cdot |E| \cdot l_{\text{max}} \cdot \log(W))\).

Then, Algorithm[7] returns the MDP \(\Gamma_{\text{MEC}}\) when the MDP \(\Gamma\) is given as input. The MECs of \(\Gamma\), that is \(\text{MEC}(\Gamma)\) can be computed in time \(O(|S| \cdot (|S| + |E|))\). Then, the most expensive operations are the successive calls to Algorithm[6] corresponding to every MEC. Since the sum of the number of states of the MECs of \(\Gamma\) is at most the number of states of \(\Gamma\), the
Algorithm 7 Replace\_MEC(Γ, l_{max})

Require: Γ = ⟨S, E, Act, s_{init}, w, P⟩ is a weighted Markov decision process and \(l_{max} \in \mathbb{N}_0\)

Ensure: \(Γ'\) is equal to \(Γ^{MEC}\)

1: \(w' = w\)
2: \(MEC\_Vect := MEC(Γ)\)
3: for \(M = (T, A) \in MEC\_Vect\) do
   4: \(λ := FixWMP\_MEC(M, l_{max})\)
   5: for \((s, a) \in T \times A\) do
      6: \(w'((s, a)) := λ\)
5: for \((s, a, s') \in E\) do
   7: \(w'((s, a, s')) := λ\)
8: \(Γ' = ⟨S, E, Act, s_{init}, w', P⟩\)
9: return \(Γ'\)

The complexity of this algorithm is in time \(O(\log(W \cdot l_{max}^2) \cdot (|S| \cdot |Act|) \cdot (|S| + |E|) \cdot l_{max} \cdot \log(W))\).

Finally, from an MDP Γ, Algorithm 8 constructs the new MDP \(Γ^{MEC}_{l_{max}}\). Then, it calls Algorithm 8.

Algorithm 8 FixWMP\_MDP(Γ, s_{init}, l_{max})

Require: Γ = ⟨S, E, Act, s_{init}, w, P⟩ is a weighted Markov decision process, \(s_{init} \in S\) and \(l_{max} \in \mathbb{N}_0\)

Ensure: \(μ\) is equal to \(E^{Γ}_{s_{init}}(f^{l_{max}}_{FixWMP})\)

1: \(Γ' = \text{Replace}\_MEC(Γ', l_{max})\)
2: \(μ := \text{ExpectedMeanPayoff}(Γ, s_{init})\)
3: return \(μ\)

Expected\_Mean\_Payoff on \(Γ^{l_{max}}_{MEC}\). Algorithm Expected\_Mean\_Payoff computes the expected value of the mean-payoff for the optimal strategy of the MDP on which it is called. For an MDP \(Γ = ⟨S, E, Act, s_{init}, w, P⟩\), it runs in time \(O(\text{Avg}(|S|, |Act|))\) (the notation \(\text{Avg}\) was introduced in the preliminaries, in Section 2.2). Therefore, the complexity of Algorithm 8 is in \(O(\log(W \cdot l_{max}^2) \cdot (|S| \cdot |Act|) \cdot (|S| + |E|) \cdot l_{max} \cdot \log(W) + \text{Avg}(|S|, |Act|))\).

Therefore, we have the following theorem:

Theorem 15. The expected value of \(f^{l_{max}}_{FixWMP}\) in an MDP Γ can be computed in time \(O(p_1(|Γ|, l_{max}))\) where \(p_1\) is a polynomial function.

4.1.2 Hardness We now consider the hardness of the problem. We show that given an MDP Γ with an initial state \(s_{init}\), a window length \(l_{max}\) and a threshold \(λ\), the fixed window mean-payoff problem of checking if \(E^{Γ}_{s_{init}}(f^{l_{max}}_{FixWMP}) \geq λ\) is at least as hard as solving the two-player game for a direct fixed window mean-payoff objective.

Theorem 16. The fixed window mean-payoff problem for MDP is at least as hard as solving a two-player game for the direct fixed window mean-payoff objective (for polynomial reductions).

Given a two-player game \(G = ⟨S_1, S_2, s_{init}, E, w⟩\), an initial state \(s_{init}\) and a window length \(l_{max}\), we construct an MDP Γ such that the expected value of the fixed window mean-payoff in Γ is equal to the outcome of the two-player game from \(s_{init}\), by using the result of Theorem 12. However, to use that theorem properly, we need the following conditions:

- The window length \(l_{max}\) is even (since, in Theorem 12 the direct fixed window mean-payoff objective considered in the two-player game has a window length equal to \(2 \cdot l_{max}\)).
• The MDP $\Gamma$ is an MEC, and therefore the two-player game is strongly connected;
• Starting the two-player game in $s_{\text{init}}$ maximizes the outcome.

Neither of these three conditions is necessarily ensured by $G$ and $l_{\text{max}}$. Therefore, we construct another two-player game $G^{\text{reset}} = (S'_1, S'_2, s_{\text{init}}, E', w')$ very similar to $G$ in which some edges are added so that, wherever the game started in $G^{\text{reset}}$, Player 2 can do as if the game started in $s_{\text{init}}$ by taking one of these edges. However, every new edges has a very high weight (equal to $(W + 1) \cdot 2 \cdot l_{\text{max}}$) so that it is not interesting for Player 2 to take more than once a new edge. In this way, the outcome of the game $G$ and $G^{\text{reset}}$ are the same from $s_{\text{init}}$. Furthermore, playing the game $G^{\text{reset}}$ from a state $s$ gives an outcome of at most what can be done from $s_{\text{init}}$.

Moreover, every edge $e$ in $G$ is duplicated to form two consecutive edges $e_1$ and $e_2$ in $G^{\text{reset}}$ with $w'(e_1) = 0$ and $w'(e_2) = w(e)$. An example of that duplication can be seen in Figure 5 and Figure 6.

In this way, the direct fixed window mean-payoff of a path in $G$ for the window length $l_{\text{max}}$ is equal to two times the direct fixed window mean-payoff of the corresponding path in $G^{\text{reset}}$ for the window length $2 \cdot l_{\text{max}}$ (this is very similar to the way a game is constructed from an MDP).

Before formally defining $G^{\text{reset}}$, we assume that the weights appearing in $G$ are all non-negative integer (w.l.o.g. since, if not, we can use the method for Markov chains described in the preliminaries, in Section 2.6).

In the following, $W$ refers to the maximum over all the weights appearing in $G$. Formally, the game $G^{\text{reset}}$ is defined such that:

- $S'_1 = S_1 \cup ((E \cap S_1 \times S_2) \times \{1\})$;
- $S'_2 = S_2 \cup ((E \cap S_1 \times S_2) \times \{2\})$;
- $E' = \{(s, (s, s', 2)) \in S'_1 \times S'_2 | (s, s') \in E \cap S_1 \times S_2\}$
  $\cup \{(s, (s, s', 1)) \in S'_2 \times S'_1 | (s, s') \in E \cap S_1 \times S_2\}$
  $\cup \{(s, (s', 1), s') \in S'_1 \times S'_2 | (s, s') \in E \cap S_1 \times S_2\}$
  $\cup E \cap S_2 \times S_1$

\[30\]
Figure 7: An example of a two-player game $G$. The circle states belong to Player 1 and the square states belong to Player 2.

Figure 8: The two-player game $G^{\text{reset}}$ obtained from the game $G$ in Figure 7 for $l_{\text{max}} = 3$. That is, $(W + 1) \cdot 2 \cdot l_{\text{max}} = 30$. The edges in $E'_5$ appear in blue.

\[ \bigcup \{(s, s', 2), s_{\text{init}}\} \in S'_2 \times S'_1 \}; \]

denoted $E'_5$.

- For $e \in E'$, we have:

\[
w'(e) = \begin{cases} 
0 & \text{if } e \in E'_1; \\
w(s, s') & \text{if } e = ((s, s'), (s, s', 1)) \in E'_2; \\
0 & \text{if } e \in E'_3; \\
w(s, s') & \text{if } e = (s, s') \in E'_4; \\
(W + 1) \cdot 2 \cdot l_{\text{max}} & \text{if } e \in E'_5 \text{ where } W = \max_{e \in E} w(e). 
\end{cases}
\]

An example of that construction can be seen in Figure 8 that is obtained from the two-player game of Figure 7.

We first establish the following two lemmas:
Lemma 17. The outcome of the two-player games $G$ and $G^{\text{reset}}$ are the same for the direct fixed window mean-payoff objective for some length $l_{\text{max}} \geq 1$. Formally:

$$
\sup_{\sigma_1 \in \text{strat}_1(G)} \inf_{\sigma_2 \in \text{strat}_2(G)} V^f_{s_{\text{init}}} (G, \sigma_1, \sigma_2) = \sup_{\sigma_1 \in \text{strat}_1(G^{\text{reset}})} \inf_{\sigma_2 \in \text{strat}_2(G^{\text{reset}})} 2 \cdot V''_{s_{\text{init}}} (G^{\text{reset}}, \sigma_1, \sigma_2)
$$

denoted $g(G, s_{\text{init}})$ and $g'(G^{\text{reset}}, s_{\text{init}})$

where $f = f_{\text{DirFixWMP}}$ and $f' = f'_{\text{DirFixWMP}}$.

Proof. By construction of the game $G^{\text{reset}}$, if Player 2 has an optimal that never takes an edge in $E'_5$, then the equality holds.

We prove that there exists an optimal strategy for Player 2 (in the game $G^{\text{reset}}$) that never takes an edge in $E'_5$. Let $\sigma'_1 \in \text{strat}_1(G^{\text{reset}})$ and $\sigma'_2 \in \text{strat}_2(G^{\text{reset}})$ be two optimal strategies for Player 1 and 2 respectively, in the game $G^{\text{reset}}$. Let $\pi = \pi(G^{\text{reset}}, s_{\text{init}}, \sigma'_1, \sigma'_2) = s_0 \ldots$ and $\lambda = f'(\pi) \leq W$. Then, there exists a sequence of $2 \cdot l_{\text{max}}$ states $\rho = s_n \ldots s_{n+2l_{\text{max}}}$ such that $\text{WMP}(\rho) = \lambda$. Then, since all the weights appearing in $G^{\text{reset}}$ are non-negative, if an edge in $E'_5$ (that is, an edge of weight $(W+1) \cdot 2 \cdot l_{\text{max}}$) was taken in that sequence, then the mean of the $2 \cdot l_{\text{max}}$ edges of that sequence would have been above $(W+1) \cdot 2 \cdot l_{\text{max}} = W + 1$. Therefore, we can conclude that no edge in $E'_5$ was taken in $\rho$.

Consider the last time an edge $e$ in $E'_5$ is taken in the path $\pi$ before visiting for the first time the sequence of states $\rho$. Then, let us denote by $\lambda' = s_{\text{init}} \ldots s_n \ldots s_{n+2l_{\text{max}}}$ the sequence of states, in $\pi$, that goes from $s_{\text{init}}$ (the destination of the edge $e$ since $e \in E'_5$) after seeing $e$. In that case, no edge in $E'_5$ is taken in $\rho'$. Therefore, the sequence of states $\rho$ is reachable from $s_{\text{init}}$ without taking any edge in $E'_5$ for Player 2.

The lemma follows.

We use the notations introduced in Lemma [17].

Lemma 18. For every state $s \in S'_1$, if we play the game starting from $s$, then the outcome of the game, that is $g(G^{\text{reset}}, s)$, is at most $g(G^{\text{reset}}, s_{\text{init}})$. That is:

$$
\max_{s \in S'_1} g(G^{\text{reset}}, s) = g(G^{\text{reset}}, s_{\text{init}})
$$

Proof. Let $\sigma_2 \in \text{strat}_2(G^{\text{reset}})$ be an optimal strategy from the state $s_{\text{init}}$. For every $s \in S'_1$, we prove that $g(G^{\text{reset}}, s) \leq g(G^{\text{reset}}, s_{\text{init}})$. We define the strategy $\sigma'_2 \in \text{strat}_2(G^{\text{reset}})$:

- $\forall s (s, s', 2) \in (S'_1 \times S'_2)$, we have $\sigma'_2(s, s', 2) = s_{\text{init}} \in S'_1$;
- $\forall s_0 \ldots s_n \in (S'_1 \times S'_2)^k$, for some $k = \frac{n+1}{2} \geq 2$, we have $\sigma'_2(s_0 \ldots s_n) = \sigma_2(s_2 \ldots s_n)$.

By definition of $\sigma'_2$, an edge in $E'_5$ is taken only once. Let $\sigma_1 \in \text{strat}_1(G^{\text{reset}})$ and let $\pi = \pi(G^{\text{reset}}, s_{\text{init}}, \sigma'_2) = s_0 \ldots$. Then, the windows opened in $s_0$ and $s_1$ are closed in $s_2 = s_{\text{init}}$ for any threshold $\lambda \leq W$ since $(s_1, s_2) = (s_1, s_{\text{init}}) \in E'_5$, that is $\lambda' = W \cdot (2 \cdot l_{\text{max}})$. Therefore, $f(\pi(G^{\text{reset}}, s_{\text{init}}, \sigma'_2)) = f(\pi(G^{\text{reset}}, s_{\text{init}}, \sigma'_2)) \leq g(G^{\text{reset}}, s_{\text{init}})$. This is true for every strategy $\sigma_1 \in \text{strat}_1(G^{\text{reset}})$. Therefore, $g(G^{\text{reset}}, s) \leq g(G^{\text{reset}}, s_{\text{init}})$. Since this is true for all $s \in S'_1$, we have $\max_{s \in S'_1} g(G^{\text{reset}}, s) = g(G^{\text{reset}}, s_{\text{init}})$.

We can proceed to the proof of Theorem [16].

Proof. Let $G = (S, S, s_{\text{init}}, E, w)$ be a two-player game and a length $l_{\text{max}} \geq 1$. We denote $G^{\text{reset}} = (S'_1, S'_2, s_{\text{init}}, E', w')$. We construct an MDP $\Gamma = (S, E, \text{Act}, s_{\text{init}}, \bar{w}, \mathbb{P})$ such that:
Figure 9: The MDP \( \Gamma \) obtained from the two-player game \( G \) in Figure 7. It ensures that \( G_\Gamma = G^{reset} \) modulo renaming of the states where \( G^{reset} \) appears in Figure 8 (recall that \( G_\Gamma \) is the two-player game obtained from the MDP \( \Gamma \)). The colors correspond to the available actions. Only in state \( s_1 \) more than one action is available.

- \( S = S'_1 \);
- \( \vec{E} \subseteq S \times Act \times S \);
- We define an action \( \alpha_s \) for every \( s \in S'_2 \). Then, \( Act = \{ \alpha_s \mid s \in S'_2 \} \cup \{ go \} \);
- For \( s \in S_1 \cup S_2 \), let \( Succ(s) = \{ s' \in S_1 \cup S_2 \mid (s, s') \in E \} \). Then, for every \( s \in S'_1 \), we have:
  \[
  Act(s) = \begin{cases} 
  \{ \alpha_{s'} \mid s' \in Succ(s) \} & \text{if } s \in S_1; \\
  \{ go \} & \text{if } s = (a, b, 1) \in S_1 \times S_2 \times \{ 1 \}
  \end{cases}
  \tag{12}
  
- For \( e = (v, \alpha, v') \in \vec{E} \), we have:
  \[
  \vec{w}(e) = \begin{cases} 
  w'(s, s', 2), (s, s', 1) & \text{if } v = s \in S, v' = (s, s', 1) \in S; \\
  w'(s', s'') & \text{if } v = (s, s', 1) \in S, v' = s'' \in S. \\
  (W + 1) \cdot 2 \cdot l_{\text{max}} & \text{if } v = s \in S, v' = s_{\text{init}} \in S;
  \end{cases}
  \tag{13}
  
- Let \( v \in S \).
  - If \( v = s \in S_1 \), then for \( \alpha_{s'} \in Act(v) \), there are two transitions: to \( (s, s', 1) \) and \( s_{\text{init}} \). Both have the same probability to occur, that is 0.5. Formally: \( P(s, \alpha_{s'}, (s, s', 1)) = \frac{1}{2} \).
  - If \( v = (s, s', 1) \in S_1 \times S_2 \times \{ 1 \} \), then only action \( go \) is available. Then, there are \( |Succ(s')| \) successor to \( v \) for that action. In fact, for every \( s'' \in Succ(s') \), we have \( P((s, s', 1), go, s'') = \frac{1}{|Succ(s')|} \).

If the probability \( P'(s, \alpha, s') \) was not defined for two states \( s, s' \in S' \), and an action \( \alpha \in Act \) then \( P'(s, \alpha, s') = 0 \). In Figure 9, we can the MDP \( \Gamma \) that was constructed from the game in Figure 7.
Now, by construction, $G_\Gamma$ (the game obtained from the MDP $\Gamma$) and $G^{\text{reset}}$ are identical modulo state renaming. Moreover, $\Gamma$ is an MEC since $G^{\text{reset}}$ is strongly connected. Then, by Theorem 12 and Lemma 18 we have that:

$$E^{\Gamma}_{s_{\text{init}}} (f_{\text{FixWMP}}) = \max_{s \in S'} g(G^{\text{reset}}, s) = g(G^{\text{reset}}, s_{\text{init}})$$

Moreover, by Lemma 17, we have:

$$g(G, s_{\text{init}}) = g(G^{\text{reset}}, s_{\text{init}})$$

In fact:

$$E^{\Gamma}_{s_{\text{init}}} (f_{\text{FixWMP}}) = g(G, s_{\text{init}}).$$

The size of $\Gamma$ is polynomial in the size of $G$. This concludes the proof.

4.2 Bounded Window Mean-Payoff

Consider a Markov decision process $\Gamma = \langle S, E, \text{Act}, s_{\text{init}}, w, P \rangle$. Then, let

$$E^{\Gamma}_{s_{\text{init}}} (f_{\text{BWMP}}) = \sup_{\sigma \in \text{strat} (\Gamma)} E^{[\sigma]}_{s_{\text{init}}} (f_{\text{BWMP}})$$

Similar to the fixed window mean-payoff, the bounded window mean-payoff is prefix independent. Therefore, we first consider the maximal end components. With the same arguments used in case of fixed window mean-payoff, if we consider an MEC $M = (T, A) \in \text{MEC}(\Gamma)$, one can argue that the expected value of the bounded window mean-payoff is the same for every state $s \in T$. This is true in every MEC $M \in \text{MEC}(\Gamma)$, hence, for every $M \in \text{MEC}(\Gamma)$, we denote by $\lambda_M$ the (optimal) expected value of $f_{\text{BWMP}}$ from every state of $M$, computed among the paths that stay in $M$, that is $\lambda_M = \max_{s \in S} E^{M}_{s} (f_{\text{BWMP}})$. In fact, the value $\lambda_M$ can be expressed using a two-player game. Formally, we have the following theorem:

**Theorem 19.** Let $M = (T, A)$ be a MEC of $\Gamma$. Then we have:

$$\lambda_M = \max_{s \in T} 2 \cdot \sup_{\sigma_1 \in \text{strat}_1 (G_M)} \inf_{\sigma_2 \in \text{strat}_2 (G_M)} V^{\text{Mean}} (G_M, \sigma_1, \sigma_2)$$

We first make a few observations.

The value $\bar{g}^s$ denotes two times the outcome of the two-player game $G_M$ resulting from $M$ where the function of interest is the mean-payoff function. From [8, 16], we know that memoryless strategies are enough to solve the two-player game for the mean-payoff objective.\(^4\)

When we construct a two-player game from an MDP, an edge of weight $w$ is split into two edges of weight 0 and $w$ (which divides the mean-payoff of a path by 2). Therefore, if the mean-payoff of a path $\pi$ in the MDP equals $2 \cdot \lambda$, then the mean-payoff of its counterpart in the two-player game equals $\lambda$. Formally, for a strategy $\sigma_1 \in \text{strat}_1 (M)$, and $\pi \in \text{Paths}_s^{M, \sigma_1}$ for some state $s \in T$, we have $2 \cdot f_{\text{Mean}} (p^s \sigma_1 (\pi)) = f_{\text{Mean}} (\pi)$. That is why $\bar{g}^s$ equals 2 times the outcome of the game.

Before proving the theorem, we first introduce the following lemma:

\(^4\)Similarly to the fixed window mean-payoff objective, since there are finitely many memoryless strategies, the supremum and infimum become maximum and minimum respectively. Also since the two-player game with bounded window objective is determined, the sup and the inf can be interchanged.
Lemma 20. Let $\mathcal{M} = \langle S, E, s_{\text{init}}, w, \mathbb{P} \rangle$ be a Markov chain and let $\pi = s_0 \ldots \in \text{Paths}^{\mathcal{M}}$. Then, we have:

$$\forall t \geq 1, \ f_{\text{FixWMP}}(t) \leq f_{\text{mean}}(t)$$

Proof. Let $l \geq 1$. We denote $f_{\text{FixWMP}}(t)$ by $\lambda_l$. By definition of $f_{\text{FixWMP}}$, we have $\pi \in \text{FixWMP}(\lambda_l, l)$. Therefore, there exists $n_l \in \mathbb{N}$ such that for all $i \geq n_l$, we have $\text{WMP}(\pi(i \ldots i+l)) \geq \lambda_l$. We prove that $f_{\text{Mean}}(\pi(n_l \ldots)) \geq \lambda_l$. Since $f_{\text{Mean}}$ is prefix independent, that would prove that $f_{\text{Mean}}(\pi) \geq \lambda_l$. Let $\epsilon > 0$. We show that there exists $n_\epsilon \in \mathbb{N}$ such that, for all $N \geq n_\epsilon$, $\frac{1}{N-n_\epsilon} \sum_{k=u_0}^{M-1} w(s_k, s_{k+1}) \geq \lambda_l - \epsilon$.

For all $i \geq n_l$, there exists $p_i \in [l]$ such that $\frac{1}{p_i} \sum_{k=0}^{p_i-1} w(s_k, s_{k+1}) \geq \lambda_l$. Let us denote by $l_i$ the smallest index satisfying this property. Then, analogous to the proof of Theorem 3, we consider the series $(u_n)_{n \in \mathbb{N}}$ defined by:

- $u_0 = n_l$;
- For all $n \geq 0$, we have $u_{n+1} = u_n + l_{u_n}$.

This series is defined such that:

- For all $n \in \mathbb{N}$, $\frac{1}{u_n-u_0} \sum_{k=u_0}^{u_n-1} w(s_k, s_{k+1}) = \frac{1}{u_n-u_0} \sum_{m=0}^{n-1} \sum_{k=u_m}^{u_{m+1}-1} w(s_k, s_{k+1}) \geq \lambda_l \frac{1}{u_n-u_0} \sum_{m=0}^{n-1} u_{m+1} - u_m$

In fact: $\frac{1}{u_n-u_0} \sum_{k=u_0}^{u_n-1} w(s_k, s_{k+1}) \geq \chi^3$

- For all $n \geq 0$, $u_{n+1} - u_n = l_{u_n} \geq 1$. Thus, for all $n \geq 0$, $u_n - u_0 \geq n$. In fact $\lim_{n \to \infty} u_n = \infty$.

Let $n_\epsilon = u_d$ where $d \in \mathbb{N}$ ensures that $\frac{d}{1+d} \geq \frac{\lambda_l - \epsilon}{\lambda_l}$ (such an integer $d$ exists since $\lim_{a \to \infty} \frac{a}{a+1} = 1 > \frac{\lambda_l - \epsilon}{\lambda_l}$). Let $N \geq n_\epsilon$. Let $t \geq d$ be such that $N = u_t + x$ with $x \leq l$. Then:

$$\frac{1}{N-u_0} \sum_{k=u_0}^{M-1} w(s_k, s_{k+1}) = \frac{1}{N-u_0} \left( \sum_{k=u_0}^{u_t-1} w(s_k, s_{k+1}) + \sum_{k=u_t}^{M-1} w(s_k, s_{k+1}) \right) \geq \frac{1}{N-u_0} \sum_{k=u_0}^{u_t-1} w(s_k, s_{k+1})$$

since all weights are positive

$$\geq \frac{u_t - u_0}{N-u_0} \lambda_l = \frac{u_t - u_0}{x+t-u_0} \lambda_l \geq \frac{t}{x+t} \lambda_l \geq \frac{d}{l+d} \lambda_l \geq \lambda_l - \epsilon$$

The last inequalities come the fact that if $a \geq b$, then $\frac{a}{a+x} \geq \frac{b}{b+x}$ since $x \geq 0$. In addition, $u_t - u_0 \geq t \geq d$ and $x \leq l$.

This holds for all $N \geq n_\epsilon$. In fact:

$$\forall \epsilon > 0, \exists n_\epsilon \in \mathbb{N} \text{ such that } \forall N \geq n_\epsilon \text{ we have } \frac{1}{N-u_0} \sum_{k=u_0}^{M-1} w(s_k, s_{k+1}) \geq \lambda_l - \epsilon$$

5This does not prove that $f_{\text{Mean}}(\pi(u_0 \ldots)) = \lim_{n \to \infty} \frac{1}{n-u_0} \sum_{k=u_0}^{n-1} w(s_k, s_{k+1}) \geq \lambda_l$. It only proves that

$$\limsup_{n \to \infty} \frac{1}{n-u_0} \sum_{k=u_0}^{n-1} w(s_k, s_{k+1}) \geq \lambda_l.$$
Proof. Let \( f \) be a state such that \( g(v) = \max_{s \in T} g(s) \). We proceed in two steps:

1. First we show that \( \lambda_M \geq g(v) \). Let \( \sigma_1 \in \text{strat}_1(G_M) \) be an optimal memoryless strategy for Player 1 from a state \( v \). Recall (from Section 2.3) that any strategy for Player 1 can be seen as a strategy in the MDP \( M \). Consider the Markov chain \( M^{[\sigma_1]} \).

   Then, \[
   E_v^M[\sigma_1](f_{\text{BWMP}}) = \sum_{B \in \text{BSCC}(M)} Pr_v(\diamond B) \cdot c_B
   \]

   where \( c_B = \min_{\rho \in \text{ElemCycle}(B)} \text{WMP} (\rho) \) (from section 3.2). We construct a strategy \( \sigma_2 \in \text{strat}_2(G_M) \) such that \( f(v, \sigma_1(\sigma_2)) \leq E_v^M[\sigma_1](f_{\text{BWMP}}) \) where the function \( f \) is defined in Equation 14.

   Let \( B' \in \text{BSCC}(M^{[\sigma_1]}) \) be such that \( Pr(\text{Reach}_v(B')) > 0 \) and:

   \[
   c_{B'} = \min_{B \in \text{BSCC}(M^{[\sigma_1]} \land \Pr_v(\diamond B) > 0)} c_B
   \]

   Let \( \rho_{B'} = s_0 \ldots s_{|\rho_{B'}|} \in \text{ElemCycle}^{B'} \) be such that \( \text{MP}(\rho_{B'}) = c_{B'} \) (recall that \( \text{MP}(\rho) \) is equal to mean of the weights appearing in \( \rho \)).

   Then, let \( \sigma_2 \in \text{strat}_2(G_M) \) be a strategy such that:

   - \( \sigma_2 \) reaches the state \( s_0 \) (which is reachable since \( s_0 \in B' \) and \( Pr_v(\diamond B) > 0 \))
   - \( \sigma_2 \) takes indefinitely the cycle \( \rho_{B'} \).

   By definition of \( \sigma_2 \), the path resulting from the two-player game \( G_M \) played with strategies \( \sigma_1 \) for Player 1 and \( \sigma_2 \) for Player 2 ensures: \( \pi_{(G_M,v,\sigma_1,\sigma_2)} = p_v^{\sigma_1}(\rho(\rho_{B'})^\omega) \) for some \( \rho \in \text{FPaths}_v(M^{[\sigma_1]}) \). Then, we have:

   \[
   2 \cdot f_{\text{Mean}}(\pi_{(G_M,v,\sigma_1,\sigma_2)}) = f_{\text{Mean}}(\rho(\rho_{B'})^\omega) = f_{\text{Mean}}((\rho_{B'})^\omega) = \text{MP}(\rho_{B'}) = c_{B'}
   \]

   Note that \( g(v) \leq 2 \cdot f_{\text{Mean}}(\pi_{(G_M,v,\sigma_1,\sigma_2)}) \) since \( g(v) \) is the value of the outcome of the two-player game when Player 2 opts for an optimal strategy. Moreover, we have:

   \[
   c_{B'} \leq \sum_{B \in \text{BSCC}(M)} Pr_v(\diamond B) \cdot c_B = E_v^M[\sigma_1](f_{\text{BWMP}})
   \]

   In addition:

   \[
   E_v^M[\sigma_1](f_{\text{BWMP}}) \leq \sup_{\sigma \in \text{strat}(M)} E_v^M[\sigma](f_{\text{BWMP}}) = E_v^M(f_{\text{BWMP}}) = \lambda_M
   \]

   Therefore, \( \lambda_M \geq E_v^M[\sigma_1](f_{\text{BWMP}}) \geq c_{B'} \geq g(v) = \max_{s \in T} g(s) \).
(2) Now, we show that $\lambda_M \leq \bar{g}(v)$. Let $\sigma_1 \in strat(M)$ be a strategy (not necessarily a memoryless one) in the MEC and let $\pi' \in Paths^M_{s_{\pi_1}}$ for some $t \in T$. Let $l \geq 1$. For every state $s$ occurring in $\pi'$, there exists a strategy $\sigma_s \in strat_2(G_M)$ such that $2 \cdot f_{\text{Mean}}(\sigma_s(s, s, \sigma_s)) \leq \bar{g}(s)$ (since $\sigma$ may not be an optimal strategy for Player 1). Let $\pi \in Paths^M_{s_{\pi_1}}$ such that $p^M_{\pi}(\pi) = \pi(G_M, s, \sigma_1, \sigma_2)$. Then, with Lemma $20$, we have $2 \cdot f^2_{\text{FixWMP}}(\pi(G_M, s, \sigma_1, \sigma_2)) = f^l_{\text{FixWMP}}(\pi) \leq f_{\text{Mean}}(\pi) = 2 \cdot f_{\text{Mean}}(\pi(G_M, s, \sigma_1, \sigma_2)).$ Therefore: $2 \cdot f^l_{\text{FixWMP}}(\pi(G_M, s, \sigma_1, \sigma_2)) \leq \bar{g}(s)$. In fact, for every state $s$ occurring in $\pi'$, there exists a strategy $\sigma_s \in strat_2(G_M)$ such that $2 \cdot f^2_{\text{FixWMP}}(\pi(G_M, s, \sigma_1, \sigma_2)) \leq \bar{g}(s)$. Then, similar to the proof of Theorem $12$, we can prove that the fixed window mean-payoff of a path for length $l$ is almost surely at most $\bar{g}(v)$. This is true for all $l \geq 1$ (since Lemma $20$ holds for all $l \geq 1$). Moreover, for all $\pi \in Paths^M_{s_{\pi_1}}$, we have:

$$f^l_{\text{BWMP}}(\pi) = \sup_{l \geq 1} f^l_{\text{FixWMP}}(\pi)$$

Therefore, almost surely, the bounded window mean-payoff of a path is at most $\bar{g}(v)$. Thus, we have $E^M_{\pi}(f_{\text{BWMP}}) \leq \bar{g}(v)$. This is true for every strategy $\sigma_1 \in strat(M)$ and for all state $t \in T$. That is, $\forall \sigma \in strat(M), \forall t \in T$ we have $E^M_{\pi}(f_{\text{BWMP}}) \leq \bar{g}(v)$. Hence

$$\lambda_M \leq \bar{g}(v) = \max_{s \in T} \bar{g}(s).$$

The theorem follows. \qed

**Corollary 20.1.** An optimal strategy for the bounded window mean-payoff objective in MDPs can be found among memoryless strategies.

**Proof.** We construct an optimal strategy in the same way we did for the fixed window mean objective (in the proof of Corollary $14.1$). It differs in the third part. We apply an optimal strategy that maximizes the outcome of the two-player bounded window mean-payoff game from the optimal state. That two-player can be solved with memoryless strategies. Since the first two steps can be done with memoryless strategies, the corollary follows. \qed

Now, to compute $E^\Gamma_{\sigma_{mean}}(f_{\text{BWMP}})$, we follow the same steps as we did for the fixed window mean-payoff objective, that is:

- We construct a new MDP $\Gamma_{B}^{\text{MEC}}$ from $\Gamma$ where every weight appearing in each MEC $M$ is replaced by $\lambda_M$.
- Then, we compute the expected mean-payoff in the new MDP $\Gamma_{B}^{\text{MEC}}$. This is equal to the expected value of the bounded window mean-payoff in $\Gamma$.

The algorithm used here are very similar to Algorithms $6$, $7$ and $8$. Algorithms $6$ is modified into Algorithm $\text{BWMP} - \text{MEC}$ where, in line 8, the algorithm called is Algorithm $\text{MeanPayoff}$ that solves the two-player game for the mean-payoff objective. Moreover, Algorithm $\text{BWMP} - \text{MEC}$ works as if the $l_{\text{max}}$ given as a parameter (that we do not need anymore, since we are in the bounded case) was equal to $|T|$ the number of states of the MEC on which it is called. That is because the value of $\lambda_M$ will be equal to the mean of an elementary cycle (as we have shown in Section $11.2$) whose length is at most $|T|$ by definition. Corresponding to $\text{Replace} - \text{MEC}$, in Algorithm $7$ we have instead $\text{BWMP} - \text{Replace} - \text{MEC}$ where in Line 4 we now call $\text{BWMP} - \text{MEC}$. Instead of $\text{FixWMP} - \text{MDP}$, in Algorithm $8$ we now have $\text{BWMP} - \text{MDP}$ that calls $\text{BWMP} - \text{Replace} - \text{MEC}$ in Line 1.

Let us denote by $\mathbb{D}$ the complexity of the $\text{MeanPayoff}$ Algorithm (recall that solving the two-player mean-payoff game is in $NP \cap coNP$). In Algorithm $6$, we enter the while loop

37
Given an MDP and a threshold $\lambda \in \mathbb{Q}$, deciding whether the expected value of $f_{BWMP}$ in the MDP is above $\lambda$ is in NP $\cap$ coNP.

### 4.2.1 Hardness

We now consider the hardness of the problem. Given an MDP $\Gamma$ with an initial state $s_{\text{init}}$ and a threshold $\lambda$, we show that the bounded window mean-payoff problem of checking if $E_{s_{\text{init}}}^\Gamma(f_{BWMP}) \geq \lambda$ is at least as hard as solving the two-player game for the mean-payoff objective.

**Theorem 22.** The bounded window mean-payoff problem for MDP is at least as hard as solving a two-player game for the mean-payoff objective (for polynomial reductions).

Given a two-player game $G = \langle S, S_1, S_2, s_{\text{init}}, w, w' \rangle$ (in which we assume that all the weight are non-negative, as we did in Section 4.1.2), consider a new two-player game $G_{B, \text{reset}} = \langle S'_1, S'_2, s_{\text{init}}, E', w' \rangle$ that is the same as $G_{\text{reset}}$ (see Section 4.1.2), except that the edges in $E'_5$ have a weight equal to $(W + 1) \cdot |S'_1 \cup S'_2|$. Considering mean-payoff objective, we have the following lemma (which is analogous to Lemma 17):

**Lemma 23.** The outcome of the two-player games $G$ and $G_{\text{reset}}$ are the same for the mean-payoff objective. Formally:

$$\inf_{\sigma_1 \in \text{strat}_1(G)} \sup_{\sigma_2 \in \text{strat}_2(G)} V^f_{s_{\text{init}}}(G, \sigma_1, \sigma_2) = \sup_{\sigma_1 \in \text{strat}_1(G_{\text{reset}})} \inf_{\sigma_2 \in \text{strat}_2(G_{\text{reset}})} 2 \cdot V^f_{s_{\text{init}}}(G_{\text{reset}}, \sigma_1, \sigma_2)$$

**Proof.** Similar to the proof of Lemma 17, we prove that there exists an optimal strategy for Player 2 that never takes an edge in $E'_5$. Let $\sigma_1 \in \text{strat}_1(G_{\text{reset}})$ and $\sigma_2 \in \text{strat}_2(G_{\text{reset}})$ be two optimal memoryless strategies for Player 1 and Player 2 respectively (see Section 4.1.2). Let $\pi = \pi(G_{\text{reset}}, s_{\text{init}}, \sigma_1, \sigma_2) = s_0 \ldots$. Assume towards a contradiction that $\sigma_2$ chooses an edge $e = (s_k, s_{k+1}) \in E'_5$ where $s_k = s_{\text{init}} = s_0$. Then, because both strategies are memoryless, the path $\pi$ is equal to $(s_0 \ldots s_k s_{k+1})^\omega$ with $k \leq |S'_1 \cup S'_2|$. Therefore, since we assume that all the weights appearing in $G$, and therefore in $G_{\text{reset}}$, are non-negative and since the weight of an edge in $E'_5$ is equal to $(W + 1) \cdot |S'_1 \cup S'_2|$, then we have (recall that the function $MP$ associates to a finite sequence of edges its mean-payoff):

$$f_{\text{mean}}(\pi) = MP(s_0 \ldots s_k s_{k+1}) \geq \frac{1}{k} w'(s_k, s_{k+1}) \geq \frac{1}{|S'_1 \cup S'_2|} (W + 1) \cdot |S'_1 \cup S'_2| = W + 1$$

Any strategy for Player 2 that never takes an edge in $E'_5$ yields an mean-payoff of at most $W$ (by definition of $W$). Hence the contradiction. That is, $\sigma_2$ never takes an edge in $E'_5$. This concludes the proof.

Then, we have the following lemma (we use the notations introduced in Lemma 23):

$$O(\log(W \cdot |T|^2))$$ times. Therefore, the complexity of Algorithm 3 is in $O(\log(W \cdot |S|^2) \cdot D)$. Then, in Algorithm 7, Algorithm 6 is called at most $|S|$ times. Therefore, the complexity of Algorithm 7 is in $O(\log(W \cdot |S|^2) \cdot D \cdot |S|)$ (since $|T| \leq |S|$ for all MEC $M = (T, A)$).

For an MDP $\Gamma = \langle S, E, Act, s_{\text{init}}, w, \mathbb{P} \rangle$, Algorithm $\text{ExpectedMeanPayoff}$ called on $\Gamma$ runs in time $O(\text{Avg}(|S|, |Act|))$. Therefore, the complexity of Algorithm 8 is in $O(\log(W \cdot |S|^2) \cdot D \cdot |S| + \text{Avg}(|S|, |Act|))$.

We have the following theorem:

**Theorem 21.** Given an MDP and a threshold $\lambda \in \mathbb{Q}$, deciding whether the expected value of $f_{BWMP}$ in the MDP is above $\lambda$ is in NP $\cap$ coNP.
Lemma 24. For every state $s \in S_1'$, if we play the game $G_B^{reset}$ starting from $s$, then the outcome of the game, that is $h(G_B^{reset}, s)$, is at most $h(G_B^{reset}, s_{init})$. That is:

$$\max_{s \in S_1'} h(G_B^{reset}, s) = h(G_B^{reset}, s_{init})$$

This lemma can be proven with the same proof we used to prove Lemma 18. Then, the proof of Theorem 22 is identical to the proof of Theorem 16 by using Lemma 23, Lemma 24, and Theorem 19. Specifically, from a two-player game $G$ and a initial state $s_{init}$, we construct an MDP $\Gamma$ such that the two-player games $G_\Gamma$ and $G_B^{reset}$ are the same modulo state renaming. Then, the expected value of the bounded window mean-payoff in $\Gamma$ is equal to the maximum (that comes from Theorem 19). Further, that maximum is achieved by starting in the state $s_{init}$ (from Lemma 24). Finally, the outcome of the two-player game $G_B^{reset}$ from $s_{init}$ for the mean-payoff function is equal to the outcome of the two-player game $G$ from $s_{init}$ (from Lemma 20). The reduction is complete.

4.3 Direct Fixed Window Mean-Payoff

Consider an MDP $\Gamma = \langle S, E, Act, s_{init}, w, P \rangle$. Let

$$E_{s_{init}}(f_{DirFixWMF}^{max}) = \sup_{\sigma \in strat(\Gamma)} E_{s_{init}}^{\Gamma[\sigma]}(f_{DirFixWMF}^{max})$$

The direct fixed window mean-payoff is not prefix independent, and therefore we cannot only look at the MECs. We present an algorithm that is exponential in $l_{max}$ and then we show the PSPACE-hardness of our problem via a reduction from the threshold problem for shortest path objectives (see [10]).

4.3.1 Algorithm

Given a weighted Markov decision process $\Gamma = \langle S, E, Act, w, P \rangle$, we construct a new MDP $\Gamma_{l_{max}} = \langle S', E', s_{init}', Act, w', P' \rangle$ with:

- $S' = S \times ([W'_0])^{l_{max}-1} \times \{ q \mid q \in [l_{max}], p \in [q \cdot W_0] \}$;
- $E' \subseteq S' \times Act \times S'$;
- $s_{init}' = (s_{init}, [W, \ldots, W], W) \in S'$, where $W = \max_{e \in E} w(e)$;
- $\forall t = (s, [a_1, \ldots, a_{l_{max}-1}], \lambda) \in S'$, we have $Act(t) = Act(s)$;
- $P'(t = (s, [a_1, \ldots, a_{l_{max}-1}], \lambda), \alpha, t' = (s', [b_1, \ldots, b_{l_{max}-1}], \lambda')) = P(s, \alpha, s')$ if $t, t' \in S'$, $\alpha \in Act$, and:
  - For all $i \in [l_{max} - 2]$, we have $b_i = a_{i+1}$;
  - $b_{l_{max}-1} = w(s, \alpha, s')$;
  - $\lambda' = \min(\lambda, \max_{l \leq l_{max}} \frac{1}{l} \sum_{k=1}^{l} a_k)$ where we denote by $a_{l_{max}}$ the number $b_{l_{max}-1}$.
- $w'(t = (s, [a_1, \ldots, a_{l_{max}-1}], \lambda), \alpha, t' = (s', [b_1, \ldots, b_{l_{max}-1}], \lambda')) = \lambda$ if $t, t' \in S'$ and $\alpha \in Act$.

Similarly to the previous constructions, if the probability $P'(s, s')$ is not defined above for two states $s, s' \in S'$, then $P'(s, s') = 0$. For convenience of notations, for a state $t = (s, [a_1, \ldots, a_{l_{max}-1}], \lambda) \in S'$, we have:
We give here the scheme of the proof, the details are provided in the appendix.

Moreover, for every path $\pi = t_0 t_1 \ldots \in \text{Paths}_{\Gamma_{\text{max}}}^{[\sigma]}(s'_{\text{init}})$ for some $\sigma \in \text{strat}(\Gamma_{\text{max}})$, we denote by $\pi^0 \in S'$ the sequence of states $t_0 t_1^0 \ldots$

Informally, the idea of this construction is the following: consider a state $t \in S'$. This state corresponds to a finite path $\rho$ in $\Gamma$ such that Last($\rho$) = $t^0$. Moreover, the last $l_{\text{max}} - 1$ weights encountered in $\rho$ are stored in $t^1$. Finally, $t^2$ keeps track of the minimum window mean-payoff seen so far in $\rho$, which is reflected on the definition of $\lambda'$. Note that the values of $t^2_0 t^2_1 \ldots$ is a non decreasing sequence.

The introduction of the new MDP $\Gamma_{\text{max}}$ is justified by the following theorem (recall that we denote by $f_{\text{mean}}$ the mean-payoff function):

**Theorem 25.** For an MDP $\Gamma$, with initial state $s_{\text{init}}$, we have:

$$E_{s_{\text{init}}}^{\Gamma_{\text{max}}} (f_{\text{DirFixWMP}}^{l_{\text{max}}}) = E_{s_{\text{init}}}^{\Gamma_{\text{max}}} (f_{\text{mean}})$$

That is, the expected value of the direct fixed window mean-payoff for the window length $l_{\text{max}}$ in $\Gamma$ is equal to the expected value of the mean-payoff in $\Gamma_{\text{max}}$.

The proof of this theorem is quite elaborate and requires the introduction of several lemmas. We give here the scheme of the proof, the details are provided in the appendix.

1. First, we introduce a function $st_1 : \text{strat}(\Gamma) \rightarrow \text{strat}(\Gamma_{\text{max}})$ that maps a strategy in $\Gamma$ to a strategy in $\Gamma_{\text{max}}$.
2. Then, we establish that, for every $\sigma \in \text{strat}(\Gamma)$, for every path $\pi \in \Gamma_{\text{max}}^{[st_1(\sigma)]}$, the sequence of states $\pi^0$ forms a path in $\Gamma^{[\sigma]}$ and the mean-payoff of $\pi$ is equal to direct fixed window mean-payoff of $\pi^0$.
3. Next, for $\sigma \in \text{strat}(\Gamma)$, we introduce a function $s^\sigma : FPaths^{\Gamma^{[\sigma]}}_{\text{init}}(s_{\text{init}}) \rightarrow S'$ that maps every finite paths in $\Gamma^\sigma$ into a state in $\Gamma_{\text{max}}$. We show that the definition of $s^\sigma$ is consistent with the definition of $\Gamma_{\text{max}}$ and $st_1$. That is, for $\sigma \in \text{strat}(\Gamma)$, if we consider a path $t_0 t_1 \ldots \in \text{Paths}_{\Gamma_{\text{max}}}^{[st_1(\sigma)]}(s'_{\text{init}})$, we have $s^\sigma(t_0 t_1^0 \ldots t_{n-1}^0 t_n) = t_n$.
4. Afterwards, for all $\sigma \in \text{strat}(\Gamma)$, we establish that there exists a bijection between the set of infinite paths of $\Gamma^{[\sigma]}$ and $\Gamma_{\text{max}}^{[st_1(\sigma)]}$ that ensures similar properties ensured by the bijection of Lemma [10] thus proving that $E_{s_{\text{init}}}^{\Gamma^{[\sigma]}} (f_{\text{DirFixWMP}}^{l_{\text{max}}}) = E_{s_{\text{init}}}^{\Gamma_{\text{max}}^{[st_1(\sigma)]}} (f_{\text{mean}})$. With this result, we are able to prove $E_{s_{\text{init}}}^{\Gamma_{\text{max}}} (f_{\text{DirFixWMP}}^{l_{\text{max}}}) \leq E_{s_{\text{init}}}^{\Gamma_{\text{max}}} (f_{\text{mean}})$.
5. Finally, we introduce a function $st_2 : \text{strat}_0(\Gamma_{\text{max}}) \rightarrow \text{strat}(\Gamma)$ (whose definition is using the function $s^\sigma$) that ensures that, for all $\sigma_0 \in \text{strat}_0(\Gamma_{\text{max}})$, we have $E_{s_{\text{init}}}^{\Gamma_{\text{max}}^{[st_1(st_2(\sigma_0))]} (f_{\text{mean}}) = E_{s_{\text{init}}}^{\Gamma_{\text{max}}^{[\sigma_0]} (f_{\text{mean}})$. From that result, it is possible to conclude that $E_{s_{\text{init}}}^{\Gamma_{\text{max}}} (f_{\text{DirFixWMP}}^{l_{\text{max}}}) \geq E_{s_{\text{init}}}^{\Gamma_{\text{max}}} (f_{\text{mean}})$.

Algorithm [9] computes $E_{s_{\text{init}}}^{\Gamma_{\text{max}}} (f_{\text{DirFixWMP}}^{l_{\text{max}}})$ using Theorem 25. The size of the MDP $\Gamma_{\text{max}}$ is in $O(|S| \cdot W_{l_{\text{max}}}^{l_{\text{max}}-1} \cdot W_{l_{\text{max}}}^{l_{\text{max}}})$. Then, Algorithm $ExpectedMeanPayoffMDP$ runs in time $O(Avg(|S| \cdot W_{l_{\text{max}}}^{l_{\text{max}}-1} |Act|))$, when called on $\Gamma_{\text{max}}$. That is also the complexity of Algorithm [9].

We have the following theorem:
acyclicity is to be interpreted over the underlying graph. \( \sigma \) is a set of target states, and for a strategy \( s \) acyclicity of that MDP implies that, from the initial state the problem asks to decide if there exists a strategy \( \lambda = E_{\text{mean payoff}}(\Gamma_{\text{max}}) \)

```
| Algorithm 9 MDPDirFixWMP(\(\Gamma, s_{\text{init}}, l_{\text{max}}\)) |
|------------------------------------------|
| **Require:** \( \Gamma = (S, E, s_{\text{init}}, Act, w, P) \) is a weighted Markov decision process, \( s_{\text{init}} \in S \) and \( l_{\text{max}} \in \mathbb{N}_0 \) |
| **Ensure:** \( \lambda \) is equal to \( \mathbb{E}_{s_{\text{init}}}^{\Gamma} (f_{\text{DirFixWMP}}^{l_{\text{max}}}) \)
| 1: \( \Gamma_{\text{max}} := (S', E', s'_{\text{init}}, Act', w', P') \)
| 2: \( \lambda := \text{ExpectedMeanPayoff}(\Gamma_{\text{max}}) \)
| 3: return \( \lambda \)
```

**Theorem 26.** The expected value of \( f_{\text{DirFixWMP}}^{l_{\text{max}}} \) in an MDP \( \Gamma \) can be computed in time \( O(p_3(|S| \cdot W^{l_{\text{max}}} \cdot l_{\text{max}}^2)) \) where \( p_3 \) is a polynomial function and \( W \) is the maximum weight appearing in the MDP \( \Gamma \).

**Theorem 27.** Given an MDP \( \Gamma \) with an initial state \( s_{\text{init}} \), a window length \( l_{\text{max}} \) and a \( \lambda \in \mathbb{Q} \), deciding whether \( \mathbb{E}_{s_{\text{init}}}^{\Gamma} (f_{\text{DirFixWMP}}^{l_{\text{max}}}) \geq \lambda \) is PSpace-Hard.

**Proof.** We show a reduction from the threshold probability problem for shortest path objectives [10]. An instance of the threshold probability problem is given by an MDP \( \Gamma = (S, E, Act, s_{\text{init}}, w, P) \) where w.l.o.g., we have that \( w \) assigns positive weights on the edges, \( T \subseteq S \) is a set of target states, and for a strategy \( \sigma \), the truncated sum \( TS^T : \text{Paths}(\Gamma[\sigma]) \rightarrow \mathbb{N} \cup \infty \) up to \( T \) from the initial state \( s_{\text{init}} \) is defined as

\[
TS^T (\rho) = \begin{cases} 
\sum_{i=0}^{n-1} w(e_i) & \text{if } \exists n \text{ such that } \rho(n) \in T \text{ and } \forall i \leq n-1, \text{ we have } \rho(i) \notin T \\
\infty & \text{if } \forall i \geq 0, \rho(i) \notin T,
\end{cases}
\]

where \( e_i = (\rho(i), a, \rho(i+1)) \), \( a \in Act \); for a threshold \( L \in \mathbb{N} \), and a probability threshold \( p \), the problem asks to decide if there exists a strategy \( \sigma \) such that \( \mathbb{P}_{s_{\text{init}}}^{\Gamma[\sigma]} ([\rho \in \text{Paths}(\Gamma[\sigma]) | TS^T (\rho) \leq L]) \geq p \). The problem is known to be PSpace-Complete, even for acyclic MDPs [10]. The target set \( T \) is assumed to be made of absorbing states (i.e., with self-loops); the acyclicity is to be interpreted over the rest of the underlying graph.

Let \( \Gamma = (S, E, Act, s_{\text{init}}, w, P) \), where \( S = T \cup V \), and \( T \) is a set of target vertices. The acyclicity of that MDP implies that, from the initial state \( s_{\text{init}} \notin T \), it takes at most \( |S| - 1 \) steps to reach a vertex in \( T \). Let \( W \) be the maximum weight appearing in \( \Gamma \). We assume that \( L \leq W \cdot (|S| - 1) \), otherwise the problem is trivial.

We construct a new MDP \( \Gamma' = (S', E', Act', s_{\text{init}}, w', P') \) where \( S' = S \cup \{s_{\text{final}_1}, s_{\text{final}_2}\} \), \( Act' = Act \cup \{\text{loop}, \alpha, \beta\} \). The set of edges \( E' = \{(v, a, s) \mid (v, a, s) \in E, v \in V, s \in S\} \cup \{(t, \alpha, s_{\text{final}_1}) \mid t \in T\} \cup \{(t, \beta, s_{\text{final}_2}) \mid t \in T\} \cup \{(s_{\text{final}_1}, \text{loop}, s_{\text{final}_1})\} \cup \{(s_{\text{final}_2}, \text{loop}, s_{\text{final}_2})\} \cup \{(v, \beta, s_{\text{final}_2}) \mid v \in V \text{ and there is no outgoing edge from } v \in \Gamma\} \). The probability function \( P' \) is defined as:

- \( P'(v, a, s) = P(v, a, s) \) such that \( (v, a, s) \in E, v \in V, s \in S \);
- \( P'(t, \alpha, s_{\text{final}_1}) = 1 \) for \( t \in T \);
- \( P'(t, \beta, s_{\text{final}_2}) = 1 \) for \( t \in T \);
- \( P'(s_{\text{final}_j}, \text{loop}, s_{\text{final}_j}) = 1 \) for \( j \in \{1, 2\} \);
- \( P'(v, \beta, s_{\text{final}_2}) = 1 \) for \( (v, \beta, s_{\text{final}_2}) \in E' \) and \( v \in V \);

The weight function \( w' \) is defined as follows.

- \( w'(v, a, s) = -w(v, a, s) \) such that \( (v, a, s) \in E, v \in V, s \in S \);
Let \( l_{\text{max}} = |S| \). Starting from \( s_{\text{init}} \), since the weights on all the edges on the paths leading to a state in \( t \in T \) are negative, the direct fixed window mean-payoff will consider paths until they reach \( s_{\text{final}} \), for \( j \in \{1, 2\} \) given that the weights on the edges outgoing from \( t \) are positive.

We now call a path to be \textit{good} if \( t \) appears in the path for some \( t \in T \), and the sum of the edges from \( s_{\text{init}} \) to \( t \) is at least \(-L\), otherwise the path is \textit{bad}. Note that for a good path, choosing \( \alpha \) leads to a direct fixed window mean-payoff of 0, while choosing \( \beta \) leads to direct fixed window mean-payoff of \(-\frac{1}{|S|}\). On the other hand, for a bad path, choosing \( \alpha \) gives a direct fixed window mean-payoff of at most \(-\frac{1}{|S|}\), while choosing \( \beta \) gives a direct fixed window mean-payoff of \(-\frac{1}{|S|}\). Therefore, for an optimal strategy, the direct fixed window mean-payoff for a \textit{good} path is 0, and for a \textit{bad} path, it is \(-\frac{1}{|S|}\).

We have \(|\Gamma'| = O(\text{poly}(|\Gamma|))\). Furthermore, the expected value of the direct fixed window mean-payoff, \( E_{\text{max}}^{\sigma} (f_{\text{DirFixWMP}}) \geq p \cdot 0 + (1 - p) \cdot -\frac{1}{|S|} = -(1 - p) \cdot \frac{1}{|S|} \) iff there is a solution to the threshold probability problem.

Note that since \( l_{\text{max}} = |S| \), deciding whether the expected value of the direct fixed window mean-payoff for an MDP is greater than or equal to some threshold is PSPACE-HARD even when \( l_{\text{max}} \) is given in unary. Thus, we cannot expect to have an algorithm that is polynomial in the value of \( l_{\text{max}} \) unless \( P = \text{PSPACE} \).

\section{Relating Fixed Window Mean-Payoff and Bounded Window Mean-Payoff}

Let \( \Pi \) be the set of strategies in an MDP \( M \). We show the following.

\begin{theorem}
\[ \sup_{\sigma \in \Pi} \left\{ E_{\text{max}}[\sigma] (f_{\text{BWMP}}) \right\} = \sup_{l_{\text{max}} \sigma \in \Pi} \left\{ E_{\text{max}}[\sigma] (f_{\text{BWMP}}) \right\}. \]
\end{theorem}

\begin{proof}
First we show in Appendix \[A \] that for every pair of sets \( A \) and \( B \) and for every function \( f \), we have that \( \sup_{a \in A} \sup_{b \in B} f(a, b) = \sup_{b \in B} \sup_{a \in A} f(a, b) \).

Thus \( \sup_{l_{\text{max}}} \left\{ E_{\text{max}}[\sigma] (f_{\text{BWMP}}) \right\} = \sup_{l_{\text{max}}} \left\{ E_{\text{max}}[\sigma] (f_{\text{BWMP}}) \right\} \). Form the definition of \( E(f_{\text{BWMP}}) \), in every Markov chain, we have \( E(f_{\text{BWMP}}) \geq \sup_{l_{\text{max}}} \left\{ E_{\text{max}}[\sigma] (f_{\text{BWMP}}) \right\} \). Thus for every strategy \( \sigma \in \Pi \), we have that \( E_{\text{max}}[\sigma] (f_{\text{BWMP}}) \geq \sup_{l_{\text{max}}} \left\{ E_{\text{max}}[\sigma] (f_{\text{BWMP}}) \right\} \). Hence \( \sup_{\sigma \in \Pi} \left\{ E_{\text{max}}[\sigma] (f_{\text{BWMP}}) \right\} \geq \sup_{\sigma \in \Pi} \left\{ E_{\text{max}}[\sigma] (f_{\text{BWMP}}) \right\} \).

For the other direction, we note that in an MDP, there exists a memoryless strategy \( \sigma_0 \) such that \( \sup_{\sigma \in \Pi} \left\{ E_{\text{max}}[\sigma] (f_{\text{BWMP}}) \right\} = E_{\text{max}}[\sigma_0] (f_{\text{BWMP}}) \). Now, from the proof of Theorem \[1 \] we have \( \sup_{\sigma \in \Pi} \left\{ E_{\text{max}}[\sigma] (f_{\text{BWMP}}) \right\} = E_{\text{max}}[\sigma_0] (f_{\text{BWMP}}) \). Hence \( \sup_{\sigma \in \Pi} \left\{ E_{\text{max}}[\sigma] (f_{\text{BWMP}}) \right\} = \sup_{\sigma \in \Pi} \left\{ E_{\text{max}}[\sigma] (f_{\text{BWMP}}) \right\}. \)
\end{proof}

\footnote{The reduction does not work for Markov chains since we cannot get a threshold for the window mean-payoff that separates the cases when there is a solution to the threshold probability problem for shortest path objective and when a solution to the problem does not exist. That is, if the sum of path from \( s_{\text{init}} \) to \( t \) is below \( L \) and the edge corresponding to action \( \alpha \) is taken in \( t \), we do not know how much below 0 will the window mean-payoff be.}
As a direct implication of the above, we also have the following proposition.

**Proposition 29.** In an MDP $M$, for every $\lambda \in \mathbb{R}$, the following holds: $\exists \sigma \in \Pi : E^M[\sigma](f_{BWMP}) > \lambda \iff \exists l_{\max}, \exists \sigma' \in \Pi : E^M[\sigma](f_{BWMP}) > \lambda$

**Proof.** For every $\lambda$, if there exists a strategy $\sigma \in \Pi$ such that $\lambda < E^M[\sigma](f_{BWMP})$, then this is equivalent to $\lambda < \sup_{\sigma \in \Pi} E^M[\sigma](f_{BWMP})$. Similarly, for every $\lambda$ if there exist an $l_{\max}$ and a $\sigma \in \Pi$, such that $\lambda < E^M[\sigma](f_{\text{FixWMP}})$, then this is equivalent to $\lambda < \sup_{l_{\max}, \sigma \in \Pi} E^M[\sigma](f_{\text{FixWMP}})$.

Now for every $\lambda$ with $\lambda < \sup_{\sigma \in \Pi} E^M[\sigma](f_{BWMP})$ we have $\lambda < \sup_{l_{\max}, \sigma \in \Pi} E^M[\sigma](f_{\text{FixWMP}})$ which we know is true by Theorem 28 and we are done.

**Remark 30.** We note that the following however is not true: For every $\lambda$, $\exists \sigma \in \Pi : E^M[\sigma](f_{BWMP}) \geq \lambda \iff \exists l_{\max}, \exists \sigma' \in \Pi : E^M[\sigma](f_{\text{FixWMP}}) \geq \lambda$.

Consider the Markov chain in Figure 10. Note that this is an MDP with a single strategy, say $\sigma$. We see that $E^M[\sigma](f_{BWMP}) = 0$ (equal to the minimum mean cycle) while for every path $\pi$ starting from $s_0$, for every $l_{\max}$, with probability 1, we have $f_{\text{FixWMP}}(\pi) < 0$ and hence $E^M[\sigma](f_{\text{FixWMP}}) < 0$. Thus the statement above is not true for $\lambda = 0$.

Also consider $\lambda \leq \sup_{l_{\max}, \sigma \in \Pi} E^M[\sigma](f_{\text{FixWMP}})$. As in the proof of Proposition 29 this is not equivalent to $\lambda \leq \exists l_{\max}, \exists \sigma' \in \Pi : E^M[\sigma](f_{\text{FixWMP}})$. In general, for some set $A$, it may be the case that $\sup_{\sigma \in A} f(\sigma) > f(a)$ for every $a \in A$.

### 6 Window Mean-Cost

Window mean-cost is similar to window mean-payoff with the difference that here the objective is to minimize the cost over a window length or to ensure that the mean-cost over a window length is less than a threshold. In particular, given an $l_{\max} \in \mathbb{N}$ and a threshold $\lambda \in \mathbb{R}$ and a position in a path over a Markov chain, we check if the window mean-cost is less than or equal to $\lambda$ over some window of length $l$ starting from that position, where $1 \leq l \leq l_{\max}$. Consider a length $l \in \mathbb{N}$ and a sequence of $l+1$ states (that is $l$ edges) $\sigma = s_0 \ldots s_l$. We first define the function $WTC$ (for Window Total cost) such that:

$$WTC(\sigma) = \min_{k \in [l]} \sum_{i=0}^{k-1} w(s_i, s_{i+1})$$

The value $WTC(\sigma)$ is the minimum total cost one can ensure over a window of length $k \in [l]$ starting from $s_0$. Similarly, we define $WMC$ (for Window mean-cost) such that:

$$WMC(\sigma) = \min_{k \in [l]} \frac{1}{k} \sum_{i=0}^{k-1} w(s_i, s_{i+1})$$

43
Basically, $\text{WMC}(\sigma)$ is the minimum mean-cost one can ensure over a window of length $k \in [l]$ starting from $s_0$. For a given path $\pi \in \text{Paths}^\Lambda$, a threshold $\lambda \in \mathbb{Q}$, a position $i \in \mathbb{N}$ and $l \leq l_{\text{max}}$, we say that the window $\pi(i \ldots (i + l))$ is closed if $\text{WMC}(\pi(i \ldots (i + l))) \leq \lambda$. Otherwise, the window is open. We note that the inductive property of windows also holds for the above definition of window mean-cost.

Let $\Lambda$ be an MC or an MDP. Given an initial state $s_{\text{init}}$ and a rational threshold $\lambda \in \mathbb{Q}$, we define the following objectives.

- Given $l_{\text{max}} \in \mathbb{N}$, the good window cost objective
  \[
  \text{GWC}(\lambda, l_{\text{max}}) = \left\{ \pi \in \text{Paths}^\Lambda \mid \text{WMC}(\pi(0 \ldots l_{\text{max}})) \leq \lambda \right\},
  \]
  requires that there exists a window starting in the first position of $\pi$ and of size at most $l_{\text{max}}$ over which the mean-cost is bounded above by the threshold $\lambda$.

- Given $l_{\text{max}} \in \mathbb{N}$, the direct fixed window mean-cost objective
  \[
  \text{DirFixWMC}(\lambda, l_{\text{max}}) = \left\{ \pi \in \text{Paths}^\Lambda \mid \forall j \geq 0, \pi(j, \infty) \in \text{GWC}(\lambda, l_{\text{max}}) \right\}
  \]
  requires that good windows of size at most $l_{\text{max}}$ exist in all positions along the play.

- The direct bounded window mean-cost objective
  \[
  \text{DirBWMC}(\lambda) = \left\{ \pi \in \text{Paths}^\Lambda \mid \exists l_{\text{max}} > 0, \pi \in \text{DirFixWMC}(\lambda, l_{\text{max}}) \right\}
  \]
  requires that there exists a bound $l_{\text{max}}$ such that the play satisfies the direct fixed objective.

- Given $l_{\text{max}} \in \mathbb{N}$, the fixed window mean-cost objective
  \[
  \text{FixWMC}(\lambda, l_{\text{max}}) = \left\{ \pi \in \text{Paths}^\Lambda \mid \exists i \in \mathbb{N}, \pi(i, \infty) \in \text{DirFixWMC}(\lambda, l_{\text{max}}) \right\}
  \]
  is the prefix-independent version of the direct fixed window objective: it requires for the existence of a suffix of the play satisfying it.

- The bounded window mean-cost objective
  \[
  \text{BWMC}(\lambda) = \left\{ \pi \in \text{Paths}^\Lambda \mid \exists l_{\text{max}} > 0, \pi \in \text{FixWMC}(\lambda, l_{\text{max}}) \right\}
  \]
  is the prefix-independent version of the direct bounded window objective.

For each of these decision problems, similar to window mean-payoff, we associate a value to every infinite path. We define the following functions, respectively for the fixed, direct fixed, bounded and direct bounded window mean-cost problem:

- \[
  f_{\text{FixWMC}}(\pi) = \inf\{\lambda \in \mathbb{Q} \mid \pi \in \text{FixWMC}(\lambda, l_{\text{max}})\}
  \]
- \[
  f_{\text{DirFixWMC}}(\pi) = \inf\{\lambda \in \mathbb{Q} \mid \pi \in \text{DirFixWMC}(\lambda, l_{\text{max}})\}
  \]
- \[
  f_{\text{DirBWMC}}(\pi) = \inf\{\lambda \in \mathbb{Q} \mid \pi \in \text{DirBWMC}(\lambda)\}
  \]
- \[
  f_{\text{BWMC}}(\pi) = \inf\{\lambda \in \mathbb{Q} \mid \pi \in \text{BWMC}(\lambda)\}
  \]

The window mean-cost can be reduced to window mean-payoff by negating all the weights on the edges, and thus for every problem considered above for window mean-payoff, a solution to the problem and taking its negation solves the corresponding problem for window mean-cost.
Problem & Complexity & Hardness
--- & --- & ---
FixWMP & Poly$(|\Gamma|, l_{\text{max}})$ & Two-player game DirFixWMP
BWMP & $\text{NP} \cap \text{coNP}$ & Two-player game MeanPayoff
DirFixWMP & Poly$(|\Gamma| \cdot W^{l_{\text{max}}}, l_{\text{max}})$ & PSPACE

Table 1: A summary of the complexity and hardness of solving different problems in Markov decision processes.

| Problem   | Complexity | Pseudopolynomial? |
|-----------|------------|-------------------|
| FixWMP    | Poly$(|M|, l_{\text{max}})$ | $l_{\text{max}}$ |
| BWMP      | Poly$(|M|)$ | - |
| DirFixWMP | Poly$(|M|, l_{\text{max}}, W)$ | $l_{\text{max}}, W$ |

Table 2: A summary of the complexity of solving different problems in Markov chains.

7 Conclusion

In this paper, we have studied in the context of MDPs the window mean-payoff objectives [7] that were originally introduced for two-player games. Those window objectives guarantee stability of the mean-payoff along outcomes contrary to the classical mean-payoff objectives. We have provided algorithms to compute the expected window mean-payoff value in MDPs for three variants: the fixed prefix independent variant which fixes the length of the window and asks for the window property to eventually hold forever, the bounded version which does not bound the window length a priori but asks for a bound to exist on each outcome, and the direct fixed window that fixes the length and asks for the property to hold from the very beginning of the outcome. The complexity of our algorithms are listed in the first column of Tab. 1. For the FixWMP problem, the complexity of our algorithm is fully polynomial when the window length is bounded polynomially in the size of the MDP, a natural assumption if we want to obtain stability of the mean-payoff over reasonable time periods.

For each of the three problems, we have also provided hardness results. The FixWMP for MDP is shown to be as hard as two player-games while BWMP is shown to be as hard as classical two-player mean-payoff games, a problem for which we do not have any polynomial time algorithm so far. Finally, we have shown that surprisingly requesting the window property to hold from the beginning of the outcome seems harder: for the DirFixWMP, we were able to prove PSPACE-HARDNESS even when the length $l_{\text{max}}$ is given in unary (while this problem has a polynomial time solution in the non direct case).

The complexity of the algorithms for Markov chains is summarized in Tab. 2.

Our definitions allow us to reduce our problems to weighted reachability problems: once an MEC is reached, the optimal value there is the value of the 2-player game from the best starting state in the MEC (see e.g. Lemma 12), or we need to reach another (better) MEC. Alternatively, there can be other definitions of window mean-payoff in which the probabilistic behavior inside the MECs also influences the optimal value. Here is a natural alternative. Given a state $s$ in an MC $M = (S, E, s_{\text{init}}, w, P)$, a window length $l_{\text{max}}$, a value threshold $\lambda$, and a probability threshold $p$, we ask whether all reachable states from $s$ (noted using CTL like syntax $\forall \Box$) ensure that, with probability at least $p$, the window mean-payoff for the next $l_{\text{max}}$ steps is at least $\lambda$. 
Formally, a state \( s \) ensures this alternative good window property, noted \( s \models \varphi_{GW}(p, l_{\text{max}}, \lambda) \), if and only if:

\[
\left[ \sum_{\pi \in \text{Paths}_{l_{\text{max}}}(s)} Pr(\pi) \text{ such that } \text{WMP}^{l_{\text{max}}}(\pi(0, \infty)) \geq \lambda \right] \geq p
\]

Then, the Markov chain \( \mathcal{M} \models \forall \square \varphi_{GW}(p, l_{\text{max}}, \lambda) \) if and only if: \( s_{\text{init}} \models \forall \square \varphi_{GW}(p, l_{\text{max}}, \lambda) \). In the case of an MDP \( \Gamma \), we ask whether there exists a strategy \( \sigma \) such that \( \Gamma[\sigma] \models \forall \square \varphi_{GW}(p, l_{\text{max}}, \lambda) \).

Unfortunately, such a definition leads to intractability even for MCs (contrary to the definitions that we have chosen).

**Theorem 31.** Given an MC \( \mathcal{M} = (S, E, s_{\text{init}}, w, \mathbb{P}) \), a window length \( l_{\text{max}} \), and a probability threshold \( p \), the problem of deciding whether \( \mathcal{M} \models \forall \square \varphi_{GW}(p, l_{\text{max}}, \lambda) \) is PP-Hard.

![Figure 11: PP-hardness for Markov chain for alternative definition of window mean-payoff](image)

**Proof.** We show a reduction from \( k \)-th largest subset which has recently been shown to be PP-complete \([11]\). The \( k \)-th largest subset problem is stated as given a finite set \( A \), a size function \( h: A \rightarrow \mathbb{N} \) assigning strictly positive integer values to elements of \( A \), and two naturals \( K, L \in \mathbb{N} \), decide if there exist \( n_B \geq K \) distinct subsets \( S_j \subseteq A \), where \( 1 \leq j \leq n_B \), such that \( \sum_{o \in S_j} h(o) \leq L \) for all these \( K \) or more subsets.

The Markov chain that is constructed from an instance \((A = \{a_1 \ldots a_n\}, h, K, L)\) of the \( K \)-th largest subset problem is shown in Figure 11. In this construction, the window length \( l_{\text{max}} = n + 1 \), the probability \( p = \frac{2n - K + 1}{2n} \) and \( a_i = h(o) \) for \( 1 \leq i \leq n \). We claim that there exists at least \( K \) subsets of sum lower than or equal to \( L \) if and only if \( s_{\text{init}} \not\models \forall \square \varphi_{GW}(p, \ell, 0) \).

Note that \( s_i \models \forall \square \varphi_{GW}(p, \ell, 0) \) for all \( i \in [n] \) since every payoff appearing on the edges on the paths starting from \( s_0 \) is non-negative. This implies that \( s_{\text{init}} \models \forall \square \varphi_{GW}(p, \ell, 0) \leftrightarrow s_{\text{init}} \models \varphi_{GW}(p, \ell, 0) \). Let \( n_B \) be the number of subsets of \( A \) of sum lower than or equal to \( L \) (which is also equal to the number of paths (out of \( 2^n \)) from \( s_0 \) to \( s_{n_B} \) of sum lower than or equal to \( L \)). Then, if we denote by \( p_B \) the probability that the window mean payoff from \( s_{\text{init}} \) is not below 0, we have that \( p_B = \frac{2n - n_B}{2n} \). It follows that: \( n_B \geq K \leftrightarrow p_B \leq \frac{2n - K}{2n} \leftrightarrow p_B < p \leftrightarrow s_{\text{init}} \not\models \varphi_{GW}(p, \ell, 0) \).

Hence, deciding if \( s_{\text{init}} \models \forall \square \varphi_{GW}(p, \ell, 0) \) holds in a Markov chain is coPP-hard (even when \( l_{\text{max}} \) is given in unary). The class PP being closed under complementation, it is also PP-HARD.

Also, in MDPs, randomized strategies are necessary.

**Theorem 32.** There exist an MDP \( \Gamma \), a probability threshold \( p \), and a window length \( l_{\text{max}} \) such that for all deterministic strategies \( \sigma \), we have \( \Gamma[\sigma] \not\models \forall \square \varphi_{GW}(p, l_{\text{max}}, \lambda) \), while there exists a randomized strategy \( \sigma' \) such that \( \Gamma[\sigma'] \models \forall \square \varphi_{GW}(p, l_{\text{max}}, \lambda) \).

**Proof.** Consider the example in Figure 12. Let \( l_{\text{max}} = 2 \). Note that the states \( s_3, s_4, s_5 \) and \( s_6 \) satisfy the property \( \forall \square \varphi_{GW}(1, 2, 0) \). When the blue action is chosen from \( s_2 \), then in the corresponding MC we have that \( s_1 \models \varphi_{GW}(0.75, 2, 0) \) corresponding to the finite paths \( s_1 s_3 s_3 \)
Figure 12: An example of an MDP where probabilities are in black and weights in red. We consider $\ell = 2$ and $p = 0.6$.

and $s_1s_2s_5$, and $s_2 \models \varphi_{GW}(0.5, 2, 0)$ for the path $s_2s_5s_5$. We call the deterministic strategy blue when the blue action is chosen from $s_2$. Thus we have $\Gamma^{[\text{blue}]} = \forall \Box \varphi_{GW}(0.5, 2, 0)$.

Now if the green action is chosen from $s_2$, then in the resulting MC we have $s_1 \models \varphi_{GW}(0.5, 2, 0)$ and $s_2 \models \varphi_{GW}(1, 2, 0)$. We call the deterministic strategy green when the green action is chosen from $s_2$. Thus we have $\Gamma^{[\text{green}]} = \forall \Box \varphi_{GW}(0.5, 2, 0)$.

Now for every randomized strategy $\sigma'$, we have $\Gamma^{[\sigma']} = \forall \Box \varphi_{GW}(p, 2, 0)$ for $p > 0.5$. In particular, let $\sigma_1$ be a randomized strategy in which the blue action is chosen with probability 0.6, and the green action with probability 0.4. Then we have $p = \min(0.75-0.6+0.5-0.4, 0.5-0.6+1-0.4) = \min(0.65, 0.7) = 0.65$. Thus $\Gamma^{[\sigma_1]} = \forall \Box \varphi_{GW}(0.65, 2, 0)$ while none of the deterministic strategies satisfies the property for $p = 0.65$. Moreover, we can show that $\max\{\{p : \Gamma^{[\sigma']} = \forall \Box \varphi_{GW}(p, 2, 0), \text{ and } \sigma' \text{ is a strategy in } \Gamma \}\} = \frac{2}{3}$ when the blue action is chosen from $s_2$ with probability $\frac{2}{3}$, and the green action is chosen with probability $\frac{1}{3}$.

References

[1] C. Baier and J-P. Katoen. Principles of model checking. MIT Press, 2008.

[2] Tomáš Brázdil, Václav Brozek, Krishnendu Chatterjee, Vojtech Forejt, and Antonín Kucera. Two views on multiple mean-payoff objectives in markov decision processes. Logical Methods in Computer Science, 10(1), 2014.

[3] Tomáš Brázdil, Krishnendu Chatterjee, Vojtech Forejt, and Antonín Kucera. Trading performance for stability in markov decision processes. J. Comput. Syst. Sci., 84:144–170, 2017.

[4] Tomáš Brázdil, Vojtech Forejt, Antonín Kucera, and Petr Novotný. Stability in graphs and games. In 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.

[5] Véronique Bruyère, Quentin Hautem, and Jean-François Raskin. On the complexity of heterogeneous multidimensional games. In 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.

[6] Krishnendu Chatterjee, Laurent Doyen, Mickael Randour, and Jean-François Raskin. Looking at mean-payoff and total-payoff through windows. In Automated Technology for Verification and Analysis - 11th International Symposium, ATVA 2013, Hanoi, Vietnam, October 15-18, 2013. Proceedings, volume 8172 of Lecture Notes in Computer Science, pages 118–132. Springer, 2013.
[7] 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.

[8] A. Ehrenfeucht and J. Mycielski. Positional strategies for mean payoff games. *International Journal of Game Theory*, 8(2):109–113, Jun 1979.

[9] Jerzy Filar and Koos Vrieze. *Competitive Markov decision processes*. Springer Science & Business Media, 2012.

[10] Christoph Haase and Stefan Kiefer. The odds of staying on budget. In *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part II*, pages 234–246, 2015.

[11] Christoph Haase and Stefan Kiefer. The complexity of the kth largest subset problem and related problems. *Inf. Process. Lett.*, 116(2):111–115, 2016.

[12] Paul Hunter, Guillermo A. Pérez, and Jean-François Raskin. Looking at mean payoff through foggy windows. *Acta Inf.*, 55(8):627–647, 2018.

[13] Richard M. Karp. A characterization of the minimum cycle mean in a digraph. *Discrete Mathematics*, 23:309–311, 1978.

[14] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994.

[15] Robert Tarjan. Depth-first search and linear graph algorithms. *SIAM journal on computing*, 1(2):146–160, 1972.

[16] Uri Zwick and Mike Paterson. The complexity of mean payoff games on graphs. *Theoretical Computer Science*, 158(1-2):343–359, 1996.
Appendix

A  Proof of $\sup_{a \in A} \sup_{b \in B} f(a, b) = \sup_{b \in B} \sup_{a \in A} f(a, b)$

For every $a \in A$, $b \in B$ and function $f$

**Lemma 33.** $\sup_{a \in A} \sup_{b \in B} f(a, b) = \sup_{b \in B} \sup_{a \in A} f(a, b)$

*Proof.* Consider $\lambda < \sup_{a \in A} \sup_{b \in B} f(a, b)$. Then there exists $b_0 \in B$ such that $\lambda < \sup_{b \in B} \sup_{a \in A} f(a, b) \leq \sup_{b \in B} f(a, b)$. Thus for every $\lambda < \sup_{a \in A} \sup_{b \in B} f(a, b)$, we have $\lambda < \sup_{b \in B} \sup_{a \in A} f(a, b)$. Hence $\sup_{a \in A} \sup_{b \in B} f(a, b) \leq \sup_{b \in B} \sup_{a \in A} f(a, b)$.

We can analogously show that $\sup_{a \in A} \sup_{b \in B} f(a, b) \leq \sup_{b \in B} \sup_{a \in A} f(a, b)$ and we are done. □

B  Proof of Lemma \[\text{33}\]

*Proof.* By definition, $f_{\text{Dir}BWMP}(\pi) \leq f_{BWMP}(\pi)$.

We now prove that $f_{BWMP}(\pi) \leq f_{\text{Dir}BWMP}(\pi)$. Let $\lambda \in \mathbb{R}$ such that $\lambda < f_{BWMP}(\pi)$. We prove that $\lambda \leq f_{\text{Dir}BWMP}(\pi)$. Let $\nu \in \mathbb{R}$ such that $\lambda < \nu < f_{BWMP}(\pi)$. Since $\nu < f_{BWMP}(\pi)$, we have $\pi \in BWMP(\nu)$. Therefore, there exists $n_\nu \in \mathbb{N}$, $l_\nu > 0$ such that, for all $j \geq n_\nu$, we have $\text{WMP}(\pi(j \ldots j + l_\nu)) = \max_{k \in [l_\nu]} \frac{1}{k} \sum_{i=0}^{k-1} w(s_{j+i}, s_{j+i+1}) \geq \nu$. Let $d = \min_{i \in [n_\nu-1]} \sum_{k=i}^{n_\nu-1} (w(s_k, s_{k+1}) - \lambda)$ be the minimum over all index $i \in [n_\nu - 1]$ of the sum of the weights, reduced by $\lambda$, from $s_i$ to $s_{n_\nu}$. Then, let $h = \lceil \frac{d}{\nu - \lambda} \rceil$ and $l_\lambda = n_\nu + l_\nu \cdot (1 + h)$.

We prove that $\pi \in \text{DirFixWMP}(\lambda, l_\lambda)$:

- For all $j \geq n_\nu$, $\pi(j, \infty) \in GW(\lambda, l_\lambda)$ since $\pi(j, \infty) \in GW(\nu, l_\nu)$, $\lambda < \nu$ and $l_\lambda \geq l_\nu$.
- Let $j < n_\nu$. We prove that there exists $l_j \in \mathbb{N}$ such that:
  - The length $l_j$ considered is at most $l_\lambda$: $l_j \leq l_\lambda$;
  - The window mean-payoff from $s_j$ to $s_{j+l_j}$ is at least $\lambda$: $\text{WMP}(\pi(j \ldots j + l_j)) \geq \lambda$.

For all $k \geq n_\nu$, there exists $p_k \in [l_\nu]$ such that $\frac{1}{p_k} \sum_{i=0}^{p_k-1} w(s_{k+i}, s_{k+i+1}) \geq \nu$. Let us denote by $k_\nu \in [l_\nu]$ the smallest index satisfying that property.

Consider the series $(u_n)_{n \in \mathbb{N}}$ defined by:

- $u_0 = n_\nu$;
- For all $n \geq 0$, we have $u_{n+1} = u_n + l_{u_n}$.

Now given some $n \in \mathbb{N}$, we partition $\pi(u_0 \ldots u_n)$ into $\pi(u_0 \ldots u_1), \ldots, \pi(u_{n-1} \ldots u_n)$, such that the mean weight over each of $\pi(u_r \ldots u_{r+1})$ for every $r \in [n-1]$ is at least $\nu$.

Formally,

$$\sum_{k=u_0}^{u_n-1} w(s_k, s_{k+1}) = \sum_{i=0}^{n-1} \sum_{k=u_i}^{u_{i+1}-1} w(s_k, s_{k+1}) \geq \nu \sum_{i=0}^{n-1} (u_{i+1} - u_i) = \nu \cdot (u_n - u_0)$$

by definition of $l_{u_i}$.
Furthermore, \( u_h - u_0 = \sum_{n=0}^{h-1} u_{n+1} - u_n = \sum_{n=0}^{h-1} l_{u_n} \). Moreover, for all \( n \geq 0, 1 \leq l_{u_n} \leq l_\nu \).

That is: \( h \leq u_h - u_0 \leq h \cdot l_\nu \).

Let \( l_j = u_h - j \). We have:

1. \( l_j = u_h - j \leq u_h = u_0 + u_h - u_0 \leq u_0 + h \cdot l_\nu \) (recall that \( l_\lambda = n_\nu + l_\nu \cdot (1 + h) \));

2. We prove that \( \text{WMP}(\pi(j \ldots j + l_j)) = \text{WMP}(\pi(j \ldots u_h)) \geq \lambda \). We have:

\[
\sum_{k=j}^{u_h-1} w(s_k, s_{k+1}) = \sum_{k=j}^{u_h-1} w(s_k, s_{k+1}) - \lambda + \lambda = \sum_{k=j}^{u_h-1} (w(s_k, s_{k+1}) - \lambda) + (u_0 - j) \cdot \lambda.
\]

Moreover, we have:

(i) \( \sum_{k=j}^{u_0-1} w(s_k, s_{k+1}) \geq d + (u_0 - j) \cdot \lambda \);

(ii) \( \sum_{k=u_0}^{u_h-1} w(s_k, s_{k+1}) \geq \nu \cdot (u_h - u_0) \), by definition of \( (u_n)_{n \in \mathbb{N}} \).

Therefore:

\[
\sum_{k=j}^{u_h-1} w(s_k, s_{k+1}) = d + (u_0 - j) \cdot \lambda + \nu \cdot (u_h - u_0) - \lambda \cdot (u_h - u_0) + \lambda \cdot (u_h - u_0)
\]

\[
= d + ((u_h - u_0) + (u_0 - j)) \cdot \lambda + (\nu - \lambda) \cdot (u_h - u_0)
\]

\[
= \lambda \cdot (u_h - j) + d + (\nu - \lambda) \cdot h \geq \lambda \cdot (u_h - j) + d + \lfloor d \rfloor \geq \lambda \cdot (u_h - j)
\]

In fact:

\[
\frac{1}{u_h - j} \sum_{k=j}^{u_h-1} w(s_k, s_{k+1}) \geq \lambda
\]

Therefore, \( \text{WMP}(\pi(j \ldots j + l_j)) = \text{WMP}(\pi(j \ldots u_h)) \geq \lambda \).

Since this is true for all \( j \geq 0 \), we have \( \pi \in \text{DirFixWMP}(\lambda, l_\lambda) \) and therefore \( \pi \in \text{DirBWMP}(\lambda) \), that is \( \lambda \leq f_{\text{DirBWMP}}(\pi) \). Since this is true for all \( \lambda < f_{\text{BWMP}}(\pi) \), we have in fact \( f_{\text{DirBWMP}}(\pi) = f_{\text{BWMP}}(\pi) \).

We note that this proof holds even if all the weights in the Markov chain are equal to any real number, not only non-negative integers.
Figure 13: The path Markov chain obtained from the weighted Markov chain of Figure 1, with $l_{max} = 2$. The label of each state is indicated in red inside of them. The probabilities are not indicated on the edges because every single one has the same: 0.5, at the exception of the two edges going inside state $s_1, s_1, s_1$, which have probability 1.

C Second Algorithm for the Direct Fixed Case in MCs

Given weighted Markov chain $M = \langle S, E, s_{init}, w, P \rangle$ we consider a structure similar to a weighted Markov chain $M_{l_{max}} = \langle S', P', s_{init}, L \rangle$ where the weights are on the states instead of the edges, and are defined using the function $L: S' \rightarrow Q$. We have:

- $S' = s_{init} \cup FPaths_{l_{max}}^M$;
- $P'(\pi, \pi') = P(s, s')$ if $\pi = s_0s_1 \ldots s_ns, \pi' = s_1 \ldots s_nss'$ and $\pi, \pi' \in FPaths_{l_{max}}^M$;
- $P'(s_{init}, \pi) = P(\pi)$ if $\pi = s_0s_1 \ldots s_n \in FPaths_{l_{max}}^M$, where $s_0 = s_{init}$;
- $L(s_{init}) = 0$;
- $L(\pi) = WMP(\pi(0 \ldots l_{max}))$, if $\pi \in S' \setminus \{s_{init}\}$.

Similarly to the previous construction, if the probability $P'(s, s')$ was not defined for two states $s, s' \in S'$, then $P'(s, s') = 0$.

An illustration of this construction can be seen in Figure 13. In this example, the new Markov chain $M_{\lambda}$ unfolds every possible paths of length $l_{max} = 2$ of the weighted Markov chain.

Let us now define $g^{l_{max}}_{\text{DirFixWMP}}: Paths_{l_{max}}^M(s_{init}) \rightarrow Q$ such that:

$$\forall \pi \in Paths_{l_{max}}^M(s_{init}), g^{l_{max}}_{\text{DirFixWMP}}(\pi) = \sup_{\lambda} \{ \lambda | \forall i \geq 1, \lambda \leq L(\pi(i)) \}$$
The expected value in the new Markov chain \( \mathcal{M}_{\text{max}} \) can be defined in a similar way that we did in \( \mathcal{M} \): 

\[
\mathbb{E}_{s_{\text{init}}}^{\mathcal{M}_{\text{max}}} (g_{\text{DirFixWMK}}^{\text{max}}) = \sum_{r \in \mathbb{Q}} r \cdot \Pr ((g_{\text{DirFixWMK}}^{\text{max}})^{-1} (\mathcal{M}_{\text{max}}, s_{\text{init}}, r))
\]

We claim that:

**Theorem 34.** For all weighted Markov chain \( \mathcal{M} \), we have:

\[
\mathbb{E}_{s_{\text{init}}}^{\mathcal{M}} (f_{\text{DirFixWMK}}^{\text{max}}) = \mathbb{E}_{s_{\text{init}}}^{\mathcal{M}_{\text{max}}} (g_{\text{DirFixWMK}}^{\text{max}})
\]

We first prove the following lemma.

**Lemma 35.** There exists a bijection \( h : \text{Paths}^{\mathcal{M}} (s_{\text{init}}) \to \text{Paths}^{\mathcal{M}_{\text{max}}} (s_{\text{init}}) \) such that:

- \( \forall \pi \in \text{Paths}^{\mathcal{M}} (s_{\text{init}}) \), \( f_{\text{DirFixWMK}}^{\text{max}} (\pi) = g_{\text{DirFixWMK}}^{\text{max}} (h(\pi)) \), that is the value of a path \( \pi \) in \( \mathcal{M} \) is the same as the value of \( h(\pi) \) in \( \mathcal{M}_{\text{max}} \);

- \( \forall j \geq l_{\text{max}}, \Pr (\pi (0 \ldots j)) = \Pr (h(\pi) (0 \ldots (j - l_{\text{max}} + 1))) \), that is any finite fragment of the path \( \pi \) has the same probability to occur in \( \mathcal{M} \) as the corresponding path fragment of \( h(\pi) \) in \( \mathcal{M}_{\text{max}} \), while considering that the \( l_{\text{max}} \) first states of \( \pi \) are merged into one state in \( \mathcal{M}_{\text{max}} \).

**Proof.** Let \( h : \text{Paths}^{\mathcal{M}} (s_{\text{init}}) \to \text{Paths}^{\mathcal{M}_{\text{max}}} (s_{\text{init}}) \) be such that for \( \pi = s_0 s_1 \ldots \in \text{Paths}^{\mathcal{M}} (s_{\text{init}}) \), where \( s_0 = s_{\text{init}} \), we have \( h(\pi) = t_0 t_1 \ldots \in \text{Paths}^{\mathcal{M}_{\text{max}}} (s_{\text{init}}) \) with \( t_0 = s_{\text{init}} \in S' \) and \( \forall i \geq 0, t_{i+1} = (s_i \ldots s_{i+l_{\text{max}}} \in S' \).

- Let us first argue that \( h \) is well defined, that is for any \( \pi \in \text{Paths}^{\mathcal{M}} (s_{\text{init}}), h(\pi) \in \text{Paths}^{\mathcal{M}_{\text{max}}} (s_{\text{init}}) \). We have that \( \Pr (t_0, t_1) > 0 \) because \( t_1 = s_0 \ldots s_{l_{\text{max}}} \in \text{FPaths}^{\mathcal{M}_{\text{max}}} (s_{\text{init}}) \), since \( t_1 \) is a prefix of \( \pi \in \text{Paths}^{\mathcal{M}} (s_{\text{init}}) \). Moreover, \( \forall i \geq 1, \Pr (t_i, t_{i+1}) = \Pr (s_{i+l_{\text{max}}-1}, s_{i+l_{\text{max}}}) > 0 \). Therefore, \( h(\pi) \in \text{Paths}^{\mathcal{M}_{\text{max}}} (s_{\text{init}}) \);

- Now we prove that if, for \( \pi, \pi' \in \text{Paths}^{\mathcal{M}} (s_{\text{init}}) \) we have \( h(\pi) = h(\pi') \), then \( \pi = \pi' \). Let \( \pi = s_0 s_1 \ldots, \pi' = s'_0 s'_1 \ldots, h(\pi) = t_0 t_1 \ldots \) and \( h(\pi') = t'_0 t'_1 \ldots \). Let \( i \in \mathbb{N} \). Since \( h(\pi) = h(\pi') \), we have \( (s_i \ldots s_{i+l_{\text{max}}}) = t_{i+1} = t'_{i+1} = (s'_i \ldots s'_{i+l_{\text{max}}}) \) and therefore \( s_i = s'_i \). In fact, \( \forall i \in \mathbb{N}, s_i = s'_i \). That is, \( \pi = \pi' \). Thus, we can conclude that \( h \) is injective;

- Let \( \pi' = t_0 t_1 \ldots \in \text{Paths}^{\mathcal{M}_{\text{max}}} (s_{\text{init}}) \). We construct a path \( \pi \in \text{Paths}^{\mathcal{M}} (s_{\text{init}}) \) such that \( h(\pi) = \pi' \) and \( \forall j \geq l_{\text{max}}, \Pr (\pi (0 \ldots j)) = \Pr (\pi' (0 \ldots (j - l_{\text{max}} + 1))) \). For all \( i \geq 1, t_i \neq s_i \) because there is no edge going to the state \( s_{\text{init}} \) in \( \mathcal{M}_{\text{max}} \), by definition of \( \mathcal{M}_{\text{max}} \). Therefore, for all \( i \geq 1 \), let \( t_i = (s_{k_{i-1}} \ldots s_{k_{i+l_{\text{max}}-1}}) \in \text{FPaths}^{\mathcal{M}_{\text{max}}} \). We can now define \( \pi = s_{k_0} s_{k_1} \ldots \) (where \( s_{k_0} = s_{\text{init}} \) since \( \pi' \in \text{Paths}^{\mathcal{M}_{\text{max}}} (s_{\text{init}}) \)). Then:

\[
- \Pr (s_{k_0} \ldots s_{k_{l_{\text{max}}}}) = \Pr (t_0, t_1) > 0
- \forall i \geq l_{\text{max}}, \Pr (s_{k_i}, s_{k_{i+1}}) = \Pr (t_{i-l_{\text{max}}+1}, t_{i-l_{\text{max}}+2}) > 0
\]

52
This implies that $\pi \in \text{Paths}^M(s_{\text{init}})$, since $\forall i \in \mathbb{N}, \mathbb{P}(s_k, s_{k+1}) > 0$. By construction, we have $h(\pi) = \pi'$. Moreover, for $j \geq l_{\text{max}}$, we have

$$\mathbb{P}(\pi(0\ldots j)) = \mathbb{P}(s_{k_0}\ldots s_{k_j}) = \mathbb{P}(s_{k_0}\ldots s_{k_{l_{\text{max}}}}) \cdot \prod_{l=l_{\text{max}}}^{j-1} \mathbb{P}(s_{k_l}, s_{k_{l+1}})$$

$$= \mathbb{P}'(t_0, t_1) \cdot \prod_{l=l_{\text{max}}}^{j-1} \mathbb{P}'(t_l, t_{l+1})$$

$$= \prod_{l=0}^{j-l_{\text{max}}} \mathbb{P}'(t_l, t_{l+1})$$

$$= \mathbb{P}'(\pi(0\ldots (j - l_{\text{max}} + 1)))$$

Thus, we can conclude that $h$ is surjective, and therefore bijective. Moreover, $h$ ensures that $\forall j \geq l_{\text{max}}, \mathbb{P}(\pi(0\ldots j)) = \mathbb{P}'(h(\pi)(0\ldots (j - l_{\text{max}} + 1)))$.

Finally, we prove that $h$ maintains the value of each path. Let $\pi = s_0 s_1 \ldots \in \text{Paths}^M(s_{\text{init}})$ such that $h(\pi) = t_0 t_1 \ldots \in \text{Paths}^M_{\text{max}}(s_{\text{init}})$. We prove that $f_{\text{DirFixWMP}}^l(\pi) = g_{\text{DirFixWMP}}^l(h(\pi))$. We have that $f_{\text{DirFixWMP}}^l(\pi) = \sup \{ \lambda \mid \pi \in \text{FixWMP}(\lambda, l_{\text{max}}) \}$. Since $\mathcal{M}$ is a finite Markov chain, there is a finite number of paths of length $l_{\text{max}}$ in $\mathcal{M}$. We can deduce that:

- $\exists i \in \mathbb{N}, f_{\text{DirFixWMP}}^l(\pi) = \text{WMP}(s_i, s_{i+l_{\text{max}}}) = L(t_{i+1})$;
- $\forall n \in \mathbb{N}, f_{\text{DirFixWMP}}^l(\pi) \leq \text{WMP}(s_n, s_{n+l_{\text{max}}}) = L(t_{n+1})$.

Since $g_{\text{DirFixWMP}}^l(h(\pi)) = \sup \{ \lambda \mid \forall i \in \mathbb{N}, \lambda \leq L(t_{i+1}) \}$, then $g_{\text{DirFixWMP}}^l(h(\pi)) = f_{\text{DirFixWMP}}^l(\pi)$. In fact, $\forall \pi \in \text{Paths}^M(s_{\text{init}})$, $f_{\text{DirFixWMP}}^l(\pi) = g_{\text{DirFixWMP}}^l(h(\pi))$.

Therefore, the function $h$ ensures the desired properties.

We can now proceed to the proof of Theorem 33.

**Proof.** Consider the function $h$ satisfying the conditions of Lemma 33. For $\lambda, \mu \in \mathbb{Q}$, let us note

$$FP^M_\lambda = \{ \pi \in \text{FPPaths}^M(s_{\text{init}}) \mid |\pi| \geq l_{\text{max}} \land \text{WMP}(\pi(|\pi| - l_{\text{max}} \ldots |\pi|)) = \lambda \land \forall i < |\pi| - l_{\text{max}}, \text{WMP}(\pi(i \ldots i + l_{\text{max}})) \neq \lambda \}$$

and

$$FP^M_{\lambda, \mu} = \{ \pi \in FP^M_\lambda \mid \exists j < |\pi| - l_{\text{max}}, \text{WMP}(\pi(|\pi| - l_{\text{max}} \ldots |\pi|)) = \mu \}$$

$$\cup \{ \pi \in FP^M_\mu \mid \exists j < |\pi| - l_{\text{max}}, \text{WMP}(\pi(|\pi| - l_{\text{max}} \ldots |\pi|)) = \lambda \}$$

Let $r \in \mathbb{Q}$. Then we have:

$$(f_{\text{DirFixWMP}}^l)^{-1}(\mathcal{M}, s_{\text{init}}, r) = \{ \pi \in \text{Paths}^M(s_{\text{init}}) \mid \exists i \in \mathbb{N}, \text{WMP}(\pi(i \ldots i + l_{\text{max}})) = r \land \forall j \in \mathbb{N}, \text{WMP}(\pi(j \ldots (j + l_{\text{max}})) \geq r) \}$$

$$= \{ \pi \in \text{Paths}^M(s_{\text{init}}) \mid \exists i \in \mathbb{N}, \text{WMP}(\pi(i \ldots (i + l_{\text{max}})) = r) \} \setminus \{ \pi \in \text{Paths}^M(s_{\text{init}}) \mid \exists i \in \mathbb{N}, \text{WMP}(\pi(i \ldots (i + l_{\text{max}})) = r \land \exists j \in \mathbb{N}, \text{WMP}(\pi(j \ldots (j + l_{\text{max}})) < r) \}$$

$$= \left( \bigcup_{\pi \in FP^M_r} \text{Cyl}(\pi) \right) \setminus \left( \bigcup_{p < r} \pi \in FP^M_{p, r} \text{Cyl}(\pi) \right)$$

53
We extend the definition of \( h \) to sets of infinite paths: for \( P \subseteq Paths^M(s_{init}) \), let \( h(P) = \{ \pi \in Paths^{M_{max}}(s_{init}) \mid \exists \pi' \in P, h(\pi') = \pi \} \). Then, by construction of \( h \), we have:

\[
(g^{l_{max}}_{\text{DirFixWMP}})^{-1}(M_{max}, s_{init}, r) = h((f^{l_{max}}_{\text{DirFixWMP}})^{-1}(M, s_{init}, r))
\]

Since all unions are disjoint in the previous series of equalities and because \( h \) is a bijection, we have:

\[
(g^{l_{max}}_{\text{DirFixWMP}})^{-1}(M_{max}, s_{init}, r) = \bigcup_{\pi \in FP^M} h(Cyl(\pi)) \setminus \bigcup_{p < r \pi \in FP^P_{p,r}} h(Cyl(\pi))
\]

Let \( \pi \in FP^M \) and \( \bar{\pi} \in Cyl(\pi) \). Then:

\[
h(Cyl(\bar{\pi})) = Cyl(h(\bar{\pi}(0\ldots|\pi| - l_{max} + 1))
\]

with \( |\pi| \geq l_{max} \) since \( \pi \in FP^M \). Therefore:

\[
Pr(h(Cyl(\bar{\pi}))) = Pr(Cyl(h(\bar{\pi}(0\ldots(|\pi| - l_{max} + 1))))
\]

Finally, we have:

\[
Pr((g^{l_{max}}_{\text{DirFixWMP}})^{-1}(M_{max}, s_{init}, r))
\]

In fact \( \forall r \in Q \):

\[
Pr((g^{l_{max}}_{\text{DirFixWMP}})^{-1}(M_{max}, s_{init}, r)) = Pr((f^{l_{max}}_{\text{DirFixWMP}})^{-1}(M, s_{init}, r))
\]

It follows that:

\[
\mathbb{E}^{M}_{s_{init}} (f^{l_{max}}_{\text{DirFixWMP}}) = \mathbb{E}^{M_{max}}_{s_{init}} (g^{l_{max}}_{\text{DirFixWMP}})
\]

Let us now focus on the computation of the expected value of \( g^{l_{max}}_{\text{DirFixWMP}} \) on a path Markov chain \( M = (S, P, s_{init}, AP, L) \). That is the purpose of Algorithm 10. The idea is that the value of a paths \( \pi \), that is \( g^{l_{max}}_{\text{DirFixWMP}}(\pi) \), depends on the value of the states visited by \( \pi \). More specifically, let us consider the set of values \( Val = \{ m \in Q \mid \exists s \in S, L(s) = m \} \) such that \( Val = \{ m_1, m_2, \ldots, m_{|Val|} \} \) with \( m_1 < m_2 < \ldots < m_{|Val|} \). Let \( \pi \in Paths^{M_{max}}(s_{init}) \). Then:

- \( g^{l_{max}}_{\text{DirFixWMP}}(\pi) = m_1 \iff \pi \models \Diamond m_1 \)
- \( g^{l_{max}}_{\text{DirFixWMP}}(\pi) = m_2 \iff \pi \models \Diamond m_2 \land \neg \Diamond m_1 \iff \pi \models (\neg m_1)U(m_2 \land \neg \Diamond m_1) \)
the following, for each 

\[ O \]

computation takes time 

\[ O \]

for weight that appears in the Markov chain, then that 

\[ W \]

whereas the previous algorithm is polynomial in 

\[ l \]

Remark 36. A possibly useful observation is that the value of path that has reached a given state can not exceed the maximum of the value of the BSCCs reachable from that state. Therefore, with some computation on the Markov chain \( M_{l_{\text{max}}} \), we could avoid some further computations in the path Markov chain.

More generally, we have:

\[ \forall i \leq |Val|, \ g_{\text{DirFixWMP}}^{l_{\text{max}}} (\pi) = m_i \Leftrightarrow \pi \models (\neg \bigvee_{j<i} m_j) U (m_i \wedge \neg \bigvee_{j<i} m_j) \]

We use the above in Algorithm 10. In the first for loop, the set \( Val \) is constructed, while associating, for each \( m \in Val \), the set of states \( S_m = \{ s \in S | L(s) = m \} \). If \( W \) be the maximal weight that appears in the Markov chain, then that for loop takes time \( O(|S| \cdot \log(W)) \). In the following, for each \( m \in Val \), the probability that \( g_{\text{DirFixWMP}}^{l_{\text{max}}} (\pi) = m \) is computed. This computation takes time \( O(\text{Mat}(S) \cdot \log(W)) \). In fact, the complexity of this algorithm is in \( O(|S| \cdot \text{Mat}(S) \cdot \log(W)) \) (since \( |Val| \leq |S| \)).

Algorithm 10 ValWindowMarkovChain(\( M, s_{\text{init}}, l_{\text{max}} \))

Require: \( M = (S, \mathcal{P}, s_{\text{init}}, AP, L) \) is a regular Markov Chain, \( s_{\text{init}} \in S \) and \( l_{\text{max}} \in \mathbb{N}_0 \)

Ensure: \( E \) is equal to \( \mathbb{E}_{\pi}^{s_{\text{init}}} (g_{\text{DirFixWMP}}^{l_{\text{max}}}) \)

1: \( Val := \text{AssocTable} (\text{float}, \text{state list}) \) :float are sorted in ascending order

2: for \( s \in S \) do

3: \quad if \( s \neq s_{\text{init}} \) then

4: \quad \quad if \( L(s) \notin Val \) then

5: \quad \quad \quad \text{Val.add}(L(s), [])

6: \quad else

7: \quad \quad \text{Val}.L(s).append(s)

8: \quad \text{GreaterVal} := []

9: \quad \text{Current} := []

10: \quad \text{E} := 0

11: for \( m \in Val \) do

12: \quad \text{Current} := \text{Val}(m)

13: \quad \text{Prob}_m := 0

14: \quad for \( s \in \text{Current} \) do

15: \quad \quad \text{C} = \text{GreaterVal} \setminus \text{Current}; \text{B} = s

16: \quad \quad \text{Prob}_m += \text{Pr}_{s_{\text{init}}} (C \cup B) \cdot \text{Pr}_s (\Box \text{GreaterVal})

17: \quad \text{GreaterVal} := \text{GreaterVal} \setminus \text{Current}

18: \quad \text{E} += \text{Prob}_m \cdot m

return \( E \)

Then, we can use the result of the previous theorem to compute the expected value of \( f_{\text{DirFixWMP}}^{l_{\text{max}}} \) in a weighted Markov chain \( M = (S, E, \mathcal{P}, w) \). That is done in algorithm 11. The size of the regular Markov chain that is constructed is in \( O(|S|^{l_{\text{max}}}) \). Therefore, the complexity of this algorithm is in \( O(|S|^{l_{\text{max}}} \cdot \text{Mat}(|S|^{l_{\text{max}}}) \cdot \log(W)) \) where \( W \) is still the maximal weight that appears in the Markov chain.

Although that complexity seems way worst that the complexity of the previous algorithm (since it is exponential in \( l_{\text{max}} \)), the main asset of this algorithm is that it is logarithmic in \( W \), whereas the previous algorithm is polynomial in \( W \).

Remark 36. A possibly useful observation is that the value of path that has reached a given state can not exceed the maximum of the value of the BSCCs reachable from that state. Therefore, with some computation on the Markov chain \( M_{l_{\text{max}}} \), we could avoid some further computations in the path Markov chain.

55
Algorithm 11 DirFixWMP2($\mathcal{M}, s_{\text{init}}, l_{\text{max}}$)

Require: $\mathcal{M} = (S, E, s_{\text{init}}, P, w)$ is a weighted Markov Chain, $s_{\text{init}} \in S$ and $l_{\text{max}} \in \mathbb{N}_0$
Ensure: $E$ is equal to $E_{\text{DirFixWMP}}^{\mathcal{M}}(f_{\text{DirFixWMP}}^{\mathcal{M}})$

1: $\mathcal{M}_{l_{\text{max}}} := (S', E', s_{\text{init}}, AP, L)$
2: $E := \text{ValWindowMarkovChain}(\mathcal{M}_{l_{\text{max}}})$
3: return $E$

D Detailed Proof of Theorem 25

We follow the five steps:

1. We introduce the function $st_1 : \text{strat}(\Gamma) \rightarrow \text{strat}(\Gamma_{l_{\text{max}}})$ such that, $\forall \sigma \in \text{strat}(\Gamma)$:

   $\forall (t_0 \ldots t_n) \in (S')^+, st_1(\sigma)(t_0 \ldots t_n) = \sigma(t_0^0 \ldots t_n^0)$

   In the following, for $\sigma \in \text{strat}(\Gamma)$, the strategy $st_1(\sigma) \in \text{strat}(\Gamma_{l_{\text{max}}})$ will be denoted $\sigma_{l_{\text{max}}}$.

   **Proposition 37.** For all $\sigma \in \text{strat}(\Gamma)$, if $\pi \in \text{Paths}_{\Gamma_{l_{\text{max}}}}^{[\sigma_{l_{\text{max}}}]}(s_{\text{init}}')$, then $\pi^0 \in \text{Paths}_{\Gamma_{l_{\text{max}}}}^{[\sigma]}(s_{\text{init}})$.

   **Proof.** This comes from the definition of $st_1$ and because, for all $t \in S'$, $\text{Act}(t) = \text{Act}(t^0)$.

2. We establish Lemma 38 that links the mean-payoff over a path $\pi$ in $\Gamma_{l_{\text{max}}}$ $[\sigma_{l_{\text{max}}}]$ and the direct fixed window mean-payoff over $\pi^0$ in $\Gamma^{[\sigma]}$.

   **Lemma 38.** Let $\sigma \in \text{strat}(\Gamma)$ and let $\pi = t_0 \ldots \in \text{Paths}_{\Gamma_{l_{\text{max}}}}^{[\sigma_{l_{\text{max}}}]}(s_{\text{init}}')$. Then $\pi^0 \in \text{Paths}_{\Gamma_{l_{\text{max}}}}^{[\sigma]}(s_{\text{init}})$ and

   \[ f_{\text{DirFixWMP}}^{\mathcal{M}}(\pi^0) = f_{\text{Mean}}(\pi) \]

   **Proof.** We have that $\pi^0 \in \text{Paths}_{\Gamma_{l_{\text{max}}}}^{[\sigma]}(s_{\text{init}})$ by Proposition 37. Therefore, $f_{\text{DirFixWMP}}^{\mathcal{M}}(\pi^0)$ is defined.

   By definition of $\Gamma_{l_{\text{max}}}$, we have: $\forall i \geq 0$, $t_i^2 \geq t_{i+1}^2$ since $t_{i+1}^2 = \min(t_i^2, x)$ where $x$ can be expressed as a function of the weights of the MDP. Moreover, for all $i \geq 0$, $w'(t_{i-1}, \sigma_{l_{\text{max}}}(t_0 \ldots t_{i-1}), t_i) = t_i^2$. Since $\Gamma_{l_{\text{max}}}$ is finite, the series $w'(t_{i-1}, \sigma_{l_{\text{max}}}(t_0 \ldots t_{i-1}), t_i)$ is a non-increasing series of rational numbers that are included in the finite set $\{ \frac{p}{q} \mid q \in [l_{\text{max}}], p \in [q \cdot W_0] \}$, therefore there exists $j \in \mathbb{N}$, and $\nu \in \mathbb{Q}$ such that:

   $\forall i \geq j$, $w'(t_i, \sigma_{l_{\text{max}}}(t_0 \ldots t_i), t_{i+1}) = \nu = t_i^2$

   It follows that $f_{\text{Mean}}(\pi) = \nu$.

   Moreover, for all $i \geq 0$, $t_i^{l_{\text{max}}-1} = w(t_0^0, \sigma_{l_{\text{max}}}(t_0 \ldots t_i), t_i^0) = w(t_{i-1}, \sigma(t_0^0 \ldots t_i^0), t_i^0)$. Hence, by definition of $\Gamma_{l_{\text{max}}}$ (specifically, the condition on $\lambda'$ so that $\mathbb{P}' > 0$) we have $t_{i_{\text{max}}}^2 = \max_{i \leq l_{\text{max}}} \frac{1}{l} \sum_{k=0}^{l-1} w(t_k^0, \sigma(t_0^0 \ldots t_k^0), t_{k+1})$. More generally:

   $\forall n \geq 0$, $t_{n+l_{\text{max}}}^2 = \min_{k \leq n} \max_{i \leq l_{\text{max}}} \frac{1}{l} \sum_{i=k}^{k+l-1} w(t_i^0, \sigma(t_0^0 \ldots t_i^0), t_{i+1})$
Then, by definition of $f_{DirFixWMPP}^{\text{max}}$, we have:

$$f_{DirFixWMPP}^{\text{max}}(\pi^0) = \inf_{n \geq 0} t_{n+l_{\text{max}}}^2 = t_f^2 = \nu = f_{\text{mean}}(\pi)$$

$\square$

3. Let $\sigma \in strat(\Gamma)$. We define by induction the function $s^\sigma : S^+ \rightarrow S'$ such that:

- $\forall s_0 \in S^0$, we have $s^\sigma(s_0) = (s_0, [W, \ldots, W], W)$;
- $\forall s_0s_1 \ldots s_n+1 \in S^n+1$, let $t_n = s^\sigma(s_0s_1 \ldots s_{n-1}s_n)$ and $t_{n+1} = s^\sigma(s_0s_1 \ldots s_ns_{n+1})$.

Then, we have:

- $\Box t_0^{n+1} = s_{n+1}$;
- for $k \leq l_{\text{max}} - 2$, we have $t_{n+k}^{(1)} = t_{n+1}^{(1)}$;
- $t_{n+1}^{(1)} = w(s_n, \sigma(s_0 \ldots s_n), s_{n+1})$;
- $t_{n+1}^{(2)} = \min(t_{n+1}^{(2)}, \max \frac{1}{f} \sum_{k=1}^{l} t^{(1)}_n)$ where $t_{n+1}^{(1)} = t_{n+1}^{(1)}$.

Let $s_0s_1 \ldots s_{n}s_{n+1} \in FPaths^{\pi(\sigma)}(s_{\text{init}})$. For all $i \leq n$, we denote $s^\sigma(s_0s_1 \ldots s_i)$ by $t_i$. Then, by construction of $s^\sigma$, for all $i \leq n$, we have:

$$\mathbb{P}'(t_i, \sigma(s_0 \ldots s_i), t_{i+1}) = \mathbb{P}(s_i, \sigma(s_0 \ldots s_i), s_{i+1}) > 0 \quad (20)$$

Note that, for a sequence of states $s_0 \ldots s_{n+1} \in S^+$, if a partial function $\iota : S^+ \rightarrow \text{Act}$ is defined on all sequence $s_0 \ldots s_i \in S^+$ for $0 \leq i \leq n$, then $s^\iota(s_0 \ldots s_n)$ is also defined (in the same way that $s^\sigma(s_0 \ldots s_n)$ is defined if $\sigma \in strat(\Gamma)$).

Moreover, $s^\sigma$ ensures the following property (recall that $\sigma_{\text{max}}$ refers to $st_1(\sigma)$):

**Lemma 39.** Let $t_0t_1 \ldots \in Paths^\Gamma_{\text{max}}[^{\sigma_{\text{max}}}](s'_{\text{init}})$. Then:

$$s^\sigma(t_0^0t_1^0 \ldots t_{n-1}^0t_n^0) = t_n$$

This property follows directly from the definition of $s^\sigma$ and $\Gamma_{\text{max}}$. Still, we give a complete proof of that property.

**Proof.** We prove the following inductive property, defined for $n \geq 0$:

$$H(n) : \forall t_0t_1 \ldots \in Paths^\Gamma_{\text{max}}[^{\sigma_{\text{max}}}](s'_{\text{init}}), s^\sigma(t_0^0t_1^0 \ldots t_{n-1}^0t_n^0) = t_n$$

- $H(0)$ is true since, for all $t_0t_1 \ldots \in Paths^\Gamma_{\text{max}}[^{\sigma_{\text{max}}}](s'_{\text{init}})$, we have $t_0 = s'_{\text{init}}$.
  Therefore $t_0^0 = s_{\text{init}}$ and $s^\sigma(t_0^0) = s^\sigma(s_{\text{init}}) = s_{\text{init}} = t_0$.
- Assume that $H(n)$ holds for some $n \geq 0$. Let $t_0t_1 \ldots \in Paths^\Gamma_{\text{max}}[^{\sigma_{\text{max}}}](s'_{\text{init}})$. We denote $t_{n+1} = s^\sigma(t_0^0t_1^0 \ldots t_{n-1}^0t_n^0)$. We prove that $t_{n+1} = t_{n+1}$. According to $H(n)$, $s^\sigma(t_{n+1}^0 \ldots t_n^0) = t_n$. By definition of $s$ and $\mathbb{P}'$ (recall that, since $t_0t_1 \ldots \in Paths^\Gamma_{\text{max}}[^{\sigma_{\text{max}}}](s'_{\text{init}})$, we have $\mathbb{P}'(t_n, \sigma_{\text{max}}(t_0 \ldots t_n), t_{n+1}) > 0)$:

  - $t_{n+1}^0 = t_{n+1}^0$;
  - for $k \leq l_{\text{max}} - 1$, we have $t_{n+k}^{(1)} = t_{n+1}^{(1)} = t_{n+1}^{(1)}$;
  - $t_{n+1}^{(1)} = w(t_0^0, \sigma_{\text{max}}(t_0 \ldots t_n), t_{n+1}^0) = t_{n+1}^{(1)}$. 

57
4. Now, we can establish a property that the function \( st_1 \) ensures:

**Lemma 40.** Let \( \sigma \in \text{strat}(\Gamma) \). There exists a bijection \( h^\sigma : \text{Paths}^\Gamma(s_{\text{init}}) \to \text{Paths}^\Gamma_{\max[\sigma_{\text{max}}]}(s'_{\text{init}}) \) such that, for all \( \pi \in \text{Paths}^\Gamma_{\max[\sigma_{\text{max}}]}(s'_{\text{init}}) \):

\[
\begin{align*}
&\text{for } \forall j \geq 0, \ P(\pi(0\ldots j)) = P'(h^\sigma(\pi)(0\ldots j)), \text{ that is any finite fragment of the path } \pi \\
&\text{has the same probability to occur in } \Gamma^{[\sigma]} \text{ as the corresponding path fragment of } h^\sigma(\pi) \text{ in } \Gamma_{\max[\sigma_{\text{max}}]}.
\end{align*}
\]

**Proof.** Let \( \sigma \in \text{strat}(\Gamma) \). Let \( h^\sigma : \text{Paths}^\Gamma_{\max[\sigma_{\text{max}}]}(s'_{\text{init}}) \to \text{Paths}^\Gamma_{\max[\sigma_{\text{max}}]}(s'_{\text{init}}) \) be such that for \( \pi = s_0 s_1 \ldots \in \text{Paths}^\Gamma_{\max[\sigma_{\text{max}}]}(s'_{\text{init}}) \), where \( s_0 = s_{\text{init}} \), we have \( h^\sigma(\pi) = t_0 t_1 \ldots \in \text{Paths}^\Gamma_{\max[\sigma_{\text{max}}]}(s'_{\text{init}}) \) with

\[
\forall i \geq 0, \ t_i = s^\sigma(s_0 \ldots s_i) \in S'\]

We have, for all \( i \geq 0 \), \( t^0_i = s_i \). Therefore, by definition of \( st_1 \), for all \( i \geq 0 \), we have \( \sigma_{\text{max}}(t_0 \ldots t_i) = \sigma(s_0 \ldots s_i) \).

Then:

- Let \( \pi \in \text{Paths}^\Gamma_{\max[\sigma_{\text{max}}]}(s'_{\text{init}}) \). By Equation 20, \( \forall i \geq 0 \), we have: \( P'(t_1, \sigma_{\text{max}}(t_0 \ldots t_i), t_{i+1}) = P'(t_1, \sigma(t_0 \ldots t_i), t_{i+1}) \). That is, \( h^\sigma \) is well defined and \( \forall j \geq 0 \) we have \( P(\pi(0\ldots j)) = P'(h^\sigma(\pi)(0\ldots j)) \).

- In the definition of \( h^\sigma \), for all \( i \geq 0 \), we have \( t^0_i = s_i \). It is easy to see that, for all \( \pi, \pi' \in \text{Paths}^\Gamma_{\max[\sigma_{\text{max}}]}(s'_{\text{init}}) \) such that \( \pi \neq \pi' \), we have \( h^\sigma(\pi) \neq h^\sigma(\pi') \). That is, \( h^\sigma \) is injective.

- Let \( \pi_{\text{max}} = t_0 t_1 \ldots \in \text{Paths}^\Gamma_{\max[\sigma_{\text{max}}]}(s'_{\text{init}}) \). We construct a path \( \pi \in \text{Paths}^\Gamma_{\max[\sigma_{\text{max}}]}(s'_{\text{init}}) \) such that \( h^\sigma(\pi) = \pi_{\text{max}} \). Let \( \pi = \pi_{\text{max}} \in \text{Paths}^\Gamma_{\max[\sigma_{\text{max}}]}(s'_{\text{init}}) \) (by Proposition 37). Moreover, \( h^\sigma(\pi) = t_0 t_1 \ldots \) where, for all \( n \geq 0 \), \( t'_n = s^\sigma(t^0_0 \ldots t^0_n) = t_n \) according to Lemma 39. That is, \( h^\sigma(\pi) = \pi_{\text{max}} \). This proves that \( h^\sigma \) is surjective, and therefore bijective.

- Let \( \pi = s_0 s_1 \ldots \in \text{Paths}^\Gamma_{\max[\sigma_{\text{max}}]}(s'_{\text{init}}) \) and let \( h^\sigma(\pi) = t_0 t_1 \ldots \in \text{Paths}^\Gamma_{\max[\sigma_{\text{max}}]}(s'_{\text{init}}) \). Then \( \pi = (h^\sigma(\pi))^0 \). Then, Lemma 35 gives us that \( f^\Gamma_{\text{DirFixWMP}}(\pi) = f_{\text{Mean}}(h^\sigma(\pi)) \).

In fact, the function \( h^\sigma \) ensures the properties specified in Lemma 40. \( \square \)
5. We now define the function \( st_2 : \text{strat}_0(\Gamma_{\text{max}}) \rightarrow \text{strat}(\Gamma) \) that maps a memoryless strategy in \( \Gamma_{\text{max}} \) into a strategy in \( \Gamma \). The function \( st_2 \) is defined by induction such that for all memoryless strategy \( \tilde{\sigma}_{\text{max}} \in \text{strat}_0(\Gamma_{\text{max}}) \):

- Let \( s_0 \in (S)^1 \). Then \( st_2(\tilde{\sigma}_{\text{max}})(s_0) = \tilde{\sigma}_{\text{max}}((s_0, [W, \ldots, W], W)) \);
- Let \( s_0 \ldots s_n \in (S)^{n+1} \). Then \( st_2(\tilde{\sigma}_{\text{max}})(s_0 \ldots s_n) = \tilde{\sigma}_{\text{max}}(s_{st_2(\tilde{\sigma}_{\text{max}})}(s_0s_1 \ldots s_n)) \). It is well defined since \( st_2(\tilde{\sigma}_{\text{max}}) \) is defined by induction and therefore \( st_2(\tilde{\sigma}_{\text{max}})(s_0s_1 \ldots s_i) \) is already defined for \( i \leq n - 1 \).

We define this function only on the set of memoryless strategies of \( \Gamma_{\text{max}} \) since we know that there exists a memoryless strategy that maximizes the expected value of the mean-payoff in \( \Gamma_{\text{max}} \).

The function \( st_2 \) is defined such that, for all \( \tilde{\sigma}_{\text{max}} \in \text{strat}_0(\Gamma_{\text{max}}) \), the strategies \( \tilde{\sigma}_{\text{max}} \) and \( st_1(st_2(\tilde{\sigma}_{\text{max}})) \) coincide on the valid paths of \( \Gamma_{\text{max}} \) and \( \Gamma_{\text{max}}[\tilde{\sigma}_{\text{max}}] \). This is stated in the following lemma:

**Lemma 41.** Let \( \tilde{\sigma}_{\text{max}} \in \text{strat}_0(\Gamma_{\text{max}}) \) and \( t_0t_1 \ldots \in \text{Paths}^{\Gamma_{\text{max}}}[\tilde{\sigma}_{\text{max}}](s_{\text{init}}) \cup \text{Paths}^{\Gamma_{\text{max}}[st_1(st_2(\tilde{\sigma}_{\text{max}}))]}(s_{\text{init}}) \). Then, for all \( n \geq 0 \), \( st_1(st_2(\tilde{\sigma}_{\text{max}}))(t_0 \ldots t_n) = \tilde{\sigma}_{\text{max}}(t_0 \ldots t_n) \).

**Proof.** Let \( \tilde{\sigma}_{\text{max}} \in \text{strat}_0(\Gamma_{\text{max}}) \). Let \( \pi = t_0t_1 \ldots \in \text{Paths}^{\Gamma_{\text{max}}}[\tilde{\sigma}_{\text{max}}](s_{\text{init}}) \) and \( \text{Paths}^{\Gamma_{\text{max}}[st_1(st_2(\tilde{\sigma}_{\text{max}}))]}(s_{\text{init}}) \). We prove the following inductive property, defined for \( n \geq 0 \):

\[
H^\pi(n) : \text{we have } t_0 \ldots t_n \in \text{FPaths}^{\Gamma_{\text{max}}[st_1(st_2(\tilde{\sigma}_{\text{max}}))]}(s_{\text{init}}) \text{ and } \tilde{\sigma}_{\text{max}}(t_0 \ldots t_n) = \tilde{\sigma}_{\text{max}}(t_0 \ldots t_n).
\]

- We have \( t_0 = s_{\text{init}} \). Hence:
  - \( t_0 \in \text{FPaths}^{\Gamma_{\text{max}}[st_1(st_2(\tilde{\sigma}_{\text{max}}))]}(s_{\text{init}}) \);
  - \( st_1(st_2(\tilde{\sigma}_{\text{max}}))(t_0) = st_2(\tilde{\sigma}_{\text{max}})(t_0) = st_2(\tilde{\sigma}_{\text{max}})(s_{\text{init}}) = \tilde{\sigma}_{\text{max}}(s_{\text{init}}) = \tilde{\sigma}_{\text{max}}(t_0) \).

Thus, \( H^\pi(0) \) holds.

- Assume that \( H^\pi(n-1) \) holds for some \( n \geq 1 \). Suppose that \( \pi \in \text{Paths}^{\Gamma_{\text{max}}[\tilde{\sigma}_{\text{max}}]}(s_{\text{init}}) \). In that case, by definition of \( \mathbb{P}^\pi \), we have:

\[
\mathbb{P}^\pi(t_{n-1}, \tilde{\sigma}_{\text{max}}(t_0 \ldots t_{n-1}), t_n) = \mathbb{P}(t_{n-1}, \tilde{\sigma}_{\text{max}}(t_0 \ldots t_{n-1}), t_n) > 0
\]

Moreover, by \( H^\pi(n-1) \), we have that:

- \( t_0 \ldots t_{n-1} \in \text{FPaths}^{\Gamma_{\text{max}}[st_1(st_2(\tilde{\sigma}_{\text{max}}))]}(s_{\text{init}}) \);
- \( \tilde{\sigma}_{\text{max}}(t_0 \ldots t_{n-1}) = st_1(st_2(\tilde{\sigma}_{\text{max}}))(t_0 \ldots t_{n-1}) \).

In fact:

\[
\mathbb{P}^\pi(t_{n-1}, st_1(st_2(\tilde{\sigma}_{\text{max}}))(t_0 \ldots t_{n-1}), t_n) = \mathbb{P}(t_{n-1}, \tilde{\sigma}_{\text{max}}(t_0 \ldots t_{n-1}), t_n) > 0
\]

Thus, \( t_0 \ldots t_n \in \text{FPaths}^{\Gamma_{\text{max}}[st_1(st_2(\tilde{\sigma}_{\text{max}}))]}(s_{\text{init}}) \). This is also the case if \( \pi \in \text{Paths}^{\Gamma_{\text{max}}[st_1(st_2(\tilde{\sigma}_{\text{max}}))]}(s_{\text{init}}) \). By Lemma [29], we have:

\[
s^{st_2(\tilde{\sigma}_{\text{max}})}(t_0 \ldots t_n) = t_n
\]
Since $\tilde{\sigma}^{l_{\text{max}}}$ is memoryless, we have $\tilde{\sigma}^{l_{\text{max}}}(t_n) = \tilde{\sigma}^{l_{\text{max}}}(t_0 \ldots t_n)$. In fact:

$$st_1(st_2(\tilde{\sigma}^{l_{\text{max}}})(t_0 \ldots t_n)) = st_2(\tilde{\sigma}^{l_{\text{max}}})(t_0 \ldots t_n) = \tilde{\sigma}^{l_{\text{max}}}(st_2(\tilde{\sigma}^{l_{\text{max}}})(t_0 \ldots t_n))$$

$$= \tilde{\sigma}^{l_{\text{max}}}(t_n) = \tilde{\sigma}^{l_{\text{max}}}(t_0 \ldots t_n)$$

Hence, $H^\pi(n + 1)$ holds.

Since this holds for every $\pi = t_0t_1 \ldots \in \text{Paths}^\Gamma_{l_{\text{max}}}([s'_{\text{init}}]) \cup \text{Paths}^\Gamma_{l_{\text{max}}}([s_1(st_2(\tilde{\sigma}^{l_{\text{max}}}))])$, the lemma follows.

We can now proceed to the proof of the Theorem 25.

Proof. We proceed in two steps:

- First we prove that $E^\Gamma_{s_{\text{init}}}(f_{\text{DirFixWMP}}^{l_{\text{max}}}) \leq E^\Gamma_{s'_{\text{init}}}(f_{\text{Mean}})$. Let $\sigma \in \text{strat}(\Gamma)$. Then, using Lemma 40, we can prove that $E^\Gamma_{s'_{\text{init}}}(f_{\text{DirFixWMP}}^{l_{\text{max}}}) = E^\Gamma_{s'_{\text{init}}}(f_{\text{Mean}})$ with the same arguments used in the proof of Theorem 34. This is true for all $\sigma \in \text{strat}(\Gamma)$. This implies that, for all $\sigma \in \text{strat}(\Gamma)$, there exists $\sigma^{l_{\text{max}}} \in \text{strat}(\Gamma_{l_{\text{max}}})$ such that $E^\Gamma_{s_{\text{init}}}(f_{\text{DirFixWMP}}^{l_{\text{max}}}) = E^\Gamma_{s'_{\text{init}}}(f_{\text{Mean}})$. Therefore:

$$E^\Gamma_{s_{\text{init}}}(f_{\text{DirFixWMP}}^{l_{\text{max}}}) \leq \sup_{\sigma \in \text{strat}(\Gamma)} E^\Gamma_{s'_{\text{init}}}(f_{\text{DirFixWMP}}^{l_{\text{max}}}) \leq E^\Gamma_{s'_{\text{init}}}(f_{\text{Mean}})$$

- Now we prove that $E^\Gamma_{s_{\text{init}}}(f_{\text{DirFixWMP}}^{l_{\text{max}}}) \geq E^\Gamma_{s'_{\text{init}}}(f_{\text{Mean}})$. There exists $\sigma^{l_{\text{max}}} \in \text{strat}(\Gamma_{l_{\text{max}}})$ such that

$$E^\Gamma_{s'_{\text{init}}}(f_{\text{Mean}}) = E^\Gamma_{s'_{\text{init}}}(f_{\text{Mean}})$$

From Lemma 41, we can conclude that the two Markov chains $\Gamma_{l_{\text{max}}}[s_1(st_2(\tilde{\sigma}^{l_{\text{max}}}))]$ and $\Gamma_{l_{\text{max}}}[s'_{\text{init}}(\tilde{\sigma}^{l_{\text{max}}})]$ are identical if we do not consider the states that are not reachable from $s'_{\text{init}}$. Therefore:

$$E^\Gamma_{s'_{\text{init}}}(f_{\text{Mean}}) = E^\Gamma_{s'_{\text{init}}}(f_{\text{Mean}})$$

Then, since $st_2(\tilde{\sigma}^{l_{\text{max}}}) \in \text{strat}(\Gamma)$, with the same arguments we used in the previous item, we can justify that:

$$E^\Gamma_{s'_{\text{init}}}(f_{\text{Mean}}) = E^\Gamma_{s'_{\text{init}}}(f_{\text{Mean}})$$

Therefore,

$$E^\Gamma_{s_{\text{init}}}(f_{\text{DirFixWMP}}^{l_{\text{max}}}) = \sup_{\sigma \in \text{strat}(\Gamma)} E^\Gamma_{s'_{\text{init}}}(f_{\text{DirFixWMP}}^{l_{\text{max}}}) \geq E^\Gamma_{s'_{\text{init}}}(f_{\text{Mean}})$$

That is, $E^\Gamma_{s_{\text{init}}}(f_{\text{DirFixWMP}}^{l_{\text{max}}}) \geq E^\Gamma_{s'_{\text{init}}}(f_{\text{Mean}})$.
In fact:

$$\mathbb{E}_{s_{init}}^{\Gamma}(f_{\text{DirFixWMP}}^{\text{max}}) = \mathbb{E}_{s_{init}}^{\Gamma_{\text{max}}}(f_{\text{Mean}})$$

This concludes the proof. \qed