Measuring Permissiveness in Parity Games: Mean-Payoff Parity Games Revisited*

Patricia Bouyer[1], Nicolas Markey[1], Jörg Olschewski[2], Michael Ummels[1,3]

[1] LSV, CNRS & ENS Cachan, France
{bouyer,markey,ummels}@lsv.ens-cachan.fr
[2] Lehrstuhl Informatik 7, RWTH Aachen University, Germany
olschewski@automata.rwth-aachen.de
[3] LAMSADE, CNRS & Université Paris-Dauphine, France

Abstract. We study nondeterministic strategies in parity games with the aim of computing a *most permissive* winning strategy. Following earlier work, we measure permissiveness in terms of the *average* number/weight of transitions blocked by a strategy. Using a translation into mean-payoff parity games, we prove that deciding (the permissiveness of) a most permissive winning strategy is in $\mathrm{NP} \cap \mathrm{coNP}$. Along the way, we provide a new study of mean-payoff parity games. In particular, we give a new algorithm for solving these games, which beats all previously known algorithms for this problem.

1 Introduction

Games extend the usual semantics of finite automata from one to several players, thus allowing to model interactions between agents acting on the progression of the automaton. This has proved very useful in computer science, especially for the formal verification of open systems interacting with their environment [20]. In this setting, the aim is to synthesise a controller under which the system behaves according to a given specification, whatever the environment does. Usually, this is modelled as a game between two players: Player 1 represents the controller and Player 2 represents the environment. The goal is then to find a *winning strategy* for Player 1, i.e. a recipe stating how the system should react to any possible action of the environment, in order to meet its specification.

In this paper, we consider *multi-strategies* (or *non-deterministic strategies*, cf. [1, 3]) as a generalisation of strategies: while strategies select only one possible action to be played in response to the behaviour of the environment, multi-strategies can retain several possible actions. Allowing several moves provides a way to cope with errors (e.g., actions being disabled for a short period, or timing imprecisions in timed games). Another quality of multi-strategies is their ability to be combined with other multi-strategies, yielding a refined multi-strategy, which is ideally winning for all of the original specifications. This offers a modular approach for solving games.

* Sponsored by ANR-06-SETI-003 DOTS, and by ESF-Eurocores LogICCC GASICS.

Classically, a strategy is more *permissive* than another one if it allows more behaviours. Under this notion, there does not need to exist a most permissive winning strategy [1]. Hence, we follow a different approach, which is of a quantitative nature: we provide a *measure* that specifies *how* permissive a given multi-strategy is. In order to do so, we consider *weighted games*, where each edge is equipped with a weight, which we treat as a *penalty* that is incurred when disallowing this edge. The penalty of a multi-strategy is then defined to be the average sum of penalties incurred in each step (in the limit). The lower this penalty is, the more permissive is the given multi-strategy. Our aim is to find one of the most permissive multi-strategies achieving a given objective.

We deal with multi-strategies by transforming a game with penalties into a *mean-payoff game* [11, 22] with classical (deterministic) strategies. A move in the latter game corresponds to a set of moves in the former, and is assigned a (negative) *reward* depending on the penalty of the original move. The penalty of a multi-strategy in the original game equals the opposite of the payoff achieved by the corresponding strategy in the mean-payoff game. In previous work, Bouyer et al. [3] introduced the notion of penalties and showed how to compute permissive strategies wrt. reachability objectives. We extend the study of [3] to parity objectives. This is a significant extension because parity objectives can express infinitary specifications. Using the above transformation, we reduce the problem of finding a most permissive strategy in a parity game with penalties to that of computing an optimal strategy in a *mean-payoff parity game*, which combines a mean-payoff objective with a parity objective.

While mean-payoff parity games have already been studied [9, 2, 7], we propose a new proof that these games are determined and that both players have optimal strategies. Moreover, we prove that the second player does not only have an optimal strategy with finite memory, but one that uses no memory at all. Finally, we provide a new algorithm for computing the values of a mean-payoff parity game, which is faster than the best known algorithms for this problem; the running time is exponential in the number of priorities and polynomial in the size of the game graph and the largest absolute weight.

In the second part of this paper, we present our results on parity games with penalties. In particular, we prove the existence of most permissive multi-strategies, and we show that the existence of a multi-strategy whose penalty is less than a given threshold can be decided in NP $\cap$ coNP. Finally, we adapt our deterministic algorithm for mean-payoff parity games to parity games with penalties. Our algorithm computes the penalties of a most permissive multi-strategy in time exponential in the number of priorities and polynomial in the size of the game graph and the largest penalty.

Due to space restrictions, most proofs are omitted in this extended abstract; they can be found in the full version of this paper [5].

Related work. Penalties as we use them were defined in [3]. Other notions of permissiveness have been defined in [1, 19], but these notions have the drawback that a most permissive strategy might not exist. Multi-strategies have also been used for different purposes in [16].

The parity condition goes back to [12, 18] and is fundamental for verification. Parity games admit optimal memoryless strategies for both players, and the problem of deciding the winner is in $\mathrm{NP} \cap \mathrm{coNP}$. As of this writing, it is not known whether parity games can be solved in polynomial time; the best known algorithms run in time polynomial in the size of the game graph but exponential in the number of priorities.

Another fundamental class of games are games with quantitative objectives. Mean-payoff games, where the aim is to maximise the average weight of the transitions taken in a play, are also in $\mathrm{NP} \cap \mathrm{coNP}$ and admit memoryless optimal strategies [11, 22]. The same is true for *energy games*, where the aim is to always keep the sum of the weights above a given threshold [6, 4]. In fact, parity games can easily be reduced to mean-payoff or energy games [13].

Finally, several game models mixing several qualitative or quantitative objectives have recently appeared in the literature: apart from mean-payoff parity games, these include generalised parity games [10], energy parity games [7] and lexicographic mean-payoff (parity) games [2] as well as generalised energy and mean-payoff games [8].

2 Preliminaries

A *weighted game graph* is a tuple $G = (Q_1, Q_2, E, \mathrm{weight})$, where $Q \coloneqq Q_1 \dot\cup Q_2$ is a finite set of *states*, $E \subseteq Q \times Q$ is the *edge* or *transition relation*, and $\mathrm{weight}\colon E \to \mathbb{R}$ is a function assigning a *weight* to every transition. When weighted game graphs are subject to algorithmic processing, we assume that these weights are integers; in this case, we set $W \coloneqq \max\{1, |\mathrm{weight}(e)| \mid e \in E\}$.

For $q \in Q$, we write qE for the set $\{q' \in Q \mid (q, q') \in E\}$ of all successors of q. We require that $qE \neq \emptyset$ for all states $q \in Q$. A subset $S \subseteq Q$ is a *subarena* of G if $qE \cap S \neq \emptyset$ for all states $q \in S$. If $S \subseteq Q$ is a subarena of G, then we can restrict G to states in S, in which case we obtain the weighted game graph $G \restriction S \coloneqq (Q_1 \cap S, Q_2 \cap S, E \cap (S \times S), \mathrm{weight} \restriction S \times S)$.

A *play* of G is an infinite sequence $\rho = \rho(0)\rho(1) \cdots \in Q^\omega$ of states such that $(\rho(i), \rho(i+1)) \in E$ for all $i \in \mathbb{N}$. We denote by $\mathrm{Out}^{\mathcal{G}}(q)$ the set of all plays ρ with $\rho(0) = q$ and by $\mathrm{Inf}(\rho)$ the set of states occurring infinitely often in ρ.

A *play prefix* or a *history* $\gamma = \gamma(0)\gamma(1) \cdots \gamma(n) \in Q^+$ is a finite, nonempty prefix of a play. For a play or a history ρ and $j < k \in \mathbb{N}$, we denote by $\rho[j, k) \coloneqq \rho[j, k-1] \coloneqq \rho(j) \cdots \rho(k-1)$ its infix starting at position j and ending at position $k - 1$.

Strategies. A *(deterministic) strategy* for Player i in G is a function $\sigma\colon Q^*Q_i \to Q$ such that $\sigma(\gamma q) \in qE$ for all $\gamma \in Q^*$ and $q \in Q_i$. A strategy σ is *memoryless* if $\sigma(\gamma q) = \sigma(q)$ for all $\gamma \in Q^*$ and $q \in Q_i$. More generally, a strategy σ is *finite-memory* if the equivalence relation $\sim \subseteq Q^* \times Q^*$, defined by $\gamma_1 \sim \gamma_2$ if and only if $\sigma(\gamma_1 \cdot \gamma) = \sigma(\gamma_2 \cdot \gamma)$ for all $\gamma \in Q^*Q_i$, has finite index.

We say that a play ρ of G is *consistent* with a strategy σ for Player i if $\rho(k+1) = \sigma(\rho[0, k])$ for all $k \in \mathbb{N}$ with $\rho(k) \in Q_i$, and denote by $\mathrm{Out}^G(\sigma, q_0)$ the set of all plays ρ of G that are consistent with σ and start in $\rho(0) = q_0$. Given a strategy σ of Player 1, a strategy τ of Player 2, and a state $q_0 \in Q$, there exists a unique play $\rho \in \mathrm{Out}^G(\sigma, q_0) \cap \mathrm{Out}^G(\tau, q_0)$, which we denote by $\rho^G(\sigma, \tau, q_0)$.

Traps and attractors. Intuitively, a set $T \subseteq Q$ of states is a *trap* for one of the two players if the other player can enforce that the play stays in this set. Formally, a trap for Player 2 (or simply a 2-trap) is a subarena $T \subseteq Q$ such that $qE \subseteq T$ for all states $q \in T \cap Q_2$, and $qE \cap T \neq \emptyset$ for all $q \in T \cap Q_1$. A trap for Player 1 (or 1-trap) is defined analogously.

If $T \subseteq Q$ is not a trap for Player 1, then Player 1 has a strategy to reach a position in $Q \setminus T$. In general, given a subset $S \subseteq Q$, we denote by $\mathrm{Attr}_1^G(S)$ the set of states from where Player 1 can force a visit to S. From every state in $\mathrm{Attr}_1^G(S)$, Player 1 has a memoryless strategy σ that guarantees a visit to S in at most $|Q|$ steps. We call the set $\mathrm{Attr}_1^G(S)$ the 1-*attractor of S* and σ an *attractor strategy for S*. The 2-*attractor* of a set S, denoted by $\mathrm{Attr}_2^G(S)$, and attractor strategies for Player 2 are defined symmetrically. Notice that for any set S, the set $Q \setminus \mathrm{Attr}_1^G(S)$ is a 1-trap, and if S is a subarena (2-trap), then $\mathrm{Attr}_1^G(S)$ is also a subarena (2-trap). Analogously, $Q \setminus \mathrm{Attr}_2^G(S)$ is a 2-trap, and if S is a subarena (1-trap), then $\mathrm{Attr}_2^G(S)$ is also a subarena (1-trap).

Convention. We often drop the superscript G from the expressions defined above, if no confusion arises, e.g. by writing $\mathrm{Out}(\sigma, q_0)$ instead of $\mathrm{Out}^G(\sigma, q_0)$.

3 Mean-payoff parity games

In this section, we establish that mean-payoff parity games are determined, that both players have optimal strategies, that for Player 2 even memoryless strategies suffice, and that the value problem for mean-payoff parity games is in NP ∩ coNP. Furthermore, we present a deterministic algorithm which computes the values in time exponential in the number of priorities, and runs in pseudo-polynomial time when the number of priorities is bounded.

Formally, a *mean-payoff parity game* is a tuple $\mathcal{G} = (G, \chi)$, where G is a weighted game graph, and $\chi\colon Q \to \mathbb{N}$ is a priority function assigning a *priority* to every state. A play $\rho = \rho(0)\rho(1)\cdots$ is *parity-winning* if the minimal priority occurring infinitely often in ρ is even, i.e., if $\min\{\chi(q) \mid q \in \mathrm{Inf}(\rho)\} \equiv 0 \pmod 2$. All notions that we have defined for weighted game graphs carry over to mean-payoff parity games. In particular, a play of $\mathcal{G}$ is just a play of G and a strategy for Player i in $\mathcal{G}$ is nothing but a strategy for Player i in G. Hence, we write $\mathrm{Out}^{\mathcal{G}}(\sigma, q)$ for $\mathrm{Out}^G(\sigma, q)$, and so on. As for weighted games graphs, we often omit the superscript if $\mathcal{G}$ is clear from the context. Finally, for a mean-payoff parity game $\mathcal{G} = (G, \chi)$ and a subarena S of G, we write $\mathcal{G} \upharpoonright S$ for the mean-payoff parity game $(G \upharpoonright S, \chi \upharpoonright S)$.

We say that a mean-payoff parity game $\mathcal{G} = (G, \chi)$ is a *mean-payoff game* if $\chi(q)$ is even for all $q \in Q$. In particular, given a weighted game graph G, we obtain a mean-payoff game by assigning priority 0 to all states. We denote this game by $(G, 0)$.

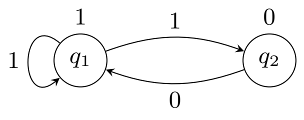

Fig. 1. A mean-payoff parity game for which infinite memory is necessary

For a play ρ of a mean-payoff parity game $\mathcal{G}$ that is parity-winning, its *payoff* is defined as

$$\mathrm{payoff}^{\mathcal{G}}(\rho) = \liminf_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} \mathrm{weight}(\rho(i), \rho(i+1))\,;$$

if ρ is not parity-winning, we set $\mathrm{payoff}^{\mathcal{G}}(\rho) := -\infty$. If σ is a strategy for Player 1 in $\mathcal{G}$, we define its *value* from $q_0 \in Q$ as $\mathrm{val}^{\mathcal{G}}(\sigma, q_0) = \inf_{\rho\in\mathrm{Out}^{\mathcal{G}}(\sigma,q_0)} \mathrm{payoff}^{\mathcal{G}}(\rho)$. Analogously, the value $\mathrm{val}^{\mathcal{G}}(\tau, q_0)$ of a strategy τ for Player 2 is defined as the supremum of $\mathrm{payoff}^{\mathcal{G}}(\rho)$ over all $\rho \in \mathrm{Out}^{\mathcal{G}}(\tau, q_0)$. The *lower* and *upper value* of a state $q_0 \in Q$ are defined by $\underline{\mathrm{val}}^{\mathcal{G}}(q_0) = \sup_\sigma \mathrm{val}^{\mathcal{G}}(\sigma, q_0)$ and $\overline{\mathrm{val}}^{\mathcal{G}}(q_0) = \inf_\tau \mathrm{val}^{\mathcal{G}}(\tau, q_0)$, respectively. Intuitively, $\underline{\mathrm{val}}^{\mathcal{G}}(q_0)$ and $\overline{\mathrm{val}}^{\mathcal{G}}(q_0)$ are the maximal (respectively minimal) payoff that Player 1 (respectively Player 2) can ensure (in the limit). We say that a strategy σ of Player 1 is *optimal from* q_0 if $\mathrm{val}^{\mathcal{G}}(\sigma, q_0) = \underline{\mathrm{val}}^{\mathcal{G}}(q_0)$. Analogously, we call a strategy τ of Player 2 optimal from q_0 if $\mathrm{val}^{\mathcal{G}}(\tau, q_0) = \overline{\mathrm{val}}^{\mathcal{G}}(q_0)$. A strategy is *(globally) optimal* if it is optimal from every state $q \in Q$. It is easy to see that $\underline{\mathrm{val}}^{\mathcal{G}}(q_0) \leq \overline{\mathrm{val}}^{\mathcal{G}}(q_0)$. If $\underline{\mathrm{val}}^{\mathcal{G}}(q_0) = \overline{\mathrm{val}}^{\mathcal{G}}(q_0)$, we say that q_0 has a *value*, which we denote by $\mathrm{val}^{\mathcal{G}}(q_0)$.

Example 1. Consider the mean-payoff parity game $\mathcal{G}$ depicted in Fig. 1, where a state or an edge is labelled with its priority, respectively weight; all states belong to Player 1. Note that $\mathrm{val}^{\mathcal{G}}(q_1) = 1$ since Player 1 can delay visiting q_2 longer and longer while still ensuring that this vertex is seen infinitely often. However, there is no finite-memory strategy that achieves this value.

It follows from Martin's determinacy theorem [17] that mean-payoff parity games are determined, i.e., that every state has a value. Moreover, Chatterjee et al. [9] gave an algorithmic proof for the existence of optimal strategies. Finally, it can be shown that for every $x \in \mathbb{R} \cup \{-\infty\}$ the set $\{\rho \in Q^\omega \mid \mathrm{payoff}(\rho) \geq x\}$ is closed under *combinations*. By Theorem 4 in [15], this property implies that Player 2 even has a memoryless optimal strategy. In the full version of this paper [5], we give a purely inductive proof of determinacy and the existence of (memoryless) optimal strategies. We thus have the following theorem.

Theorem 2. *Let $\mathcal{G}$ be a mean-payoff parity game.*

1. *$\mathcal{G}$ is determined;*
2. *Player 1 has an optimal strategy in $\mathcal{G}$;*
3. *Player 2 has a memoryless optimal strategy in $\mathcal{G}$.*

A consequence of the proof of Theorem 2 is that each value of a mean-payoff parity game is either $-\infty$ or equals one of the values of a mean-payoff game played on the same weighted graph (or a subarena of it). Since optimal memoryless strategies exist in mean-payoff games [11], the values of a mean-payoff game with integral weights are rational numbers of the form r/s with $|r| \leq |Q| \cdot W$ and $|s| \leq |Q|$. Consequently, this property holds for the (finite) values of a mean-payoff parity game as well.

We now turn towards the computational complexity of mean-payoff parity games. Formally, the *value problem* is the following decision problem: Given a mean-payoff parity game $\mathcal{G}$ (with integral weights), a designated state $q_0 \in Q$, and a number $x \in \mathbb{Q}$, decide whether $\mathrm{val}^{\mathcal{G}}(q_0) \geq x$. By Theorem 2, to decide whether $\mathrm{val}^{\mathcal{G}}(q_0) < x$, we can guess a memoryless strategy τ for Player 2 and check whether $\mathrm{val}^{\mathcal{G}}(\tau, q_0) < x$. It follows from a result of Karp [14] that the latter check can be carried out in polynomial time. Hence, the value problem belongs to coNP.

Corollary 3. *The value problem for mean-payoff parity games is in coNP.*

Via a reduction to *energy parity games*, Chatterjee and Doyen [7] recently proved that the value problem for mean-payoff parity games is in NP. Hence, these games do not seem harder than parity or mean-payoff games, which also come with a value problem in NP $\cap$ coNP.

Theorem 4 (Chatterjee-Doyen). *The value problem for mean-payoff parity games is in NP.*

A deterministic algorithm. We now present a deterministic algorithm for computing the values of a mean-payoff parity game, which runs faster than all known algorithms for solving these games. Algorithm SolveMPP is based on the classical algorithm for solving parity games, due to Zielonka [21]. The algorithm employs as a subprocedure an algorithm SolveMP for solving mean-payoff games. By [22], such an algorithm can be implemented to run in time $\mathrm{O}(n^3 \cdot m \cdot W)$ for a game with n states and m edges. We denote by $f \sqcup g$ and $f \sqcap g$ the pointwise maximum, respectively minimum, of two (partial) functions $f, g \colon Q \to \mathbb{R} \cup \{\pm\infty\}$ (where $(f \sqcup g)(q) = (f \sqcap g)(q) = f(q)$ if $g(q)$ is undefined).

The algorithm works as follows: If the least priority p in $\mathcal{G}$ is even, the algorithm first identifies the least value of $\mathcal{G}$ by computing the values of the mean-payoff game $(G, 0)$ and (recursively) the values of the game $\mathcal{G} \restriction Q \setminus \mathrm{Attr}_1(\chi^{-1}(p))$, and taking their minimum x. All states from where Player 2 can enforce a visit to a state with value x in one of these two games must have value x in $\mathcal{G}$. In the remaining subarena, the values can be computed by calling SolveMPP recursively. If the least priority is odd, we can similarly compute the greatest value of $\mathcal{G}$ and proceed by recursion. The correctness of the algorithm relies on the following two lemmas.

Lemma 5. *Let $\mathcal{G}$ be a mean-payoff parity game with least priority p even, $T = Q \setminus \mathrm{Attr}_1(\chi^{-1}(p))$, and $x \in \mathbb{R}$. If $\mathrm{val}^{(G,0)}(q) \geq x$ for all $q \in Q$ and $\mathrm{val}^{\mathcal{G} \restriction T}(q) \geq x$ for all $q \in T$, then $\mathrm{val}^{\mathcal{G}}(q) \geq x$ for all $q \in Q$.*

Algorithm SolveMPP($\mathcal{G}$)

```
Input: mean-payoff parity game $\mathcal{G} = (G, \chi)$
Output: $\mathrm{val}^{\mathcal{G}}$

if $Q = \emptyset$ then return $\emptyset$
$p \coloneqq \min\{\chi(q) \mid q \in Q\}$
if $p$ is even then
    $g \coloneqq \mathrm{SolveMP}(G, 0)$
    if $\chi(q) = p$ for all $q \in Q$ then return $g$
    $T \coloneqq Q \setminus \mathrm{Attr}_1^{\mathcal{G}}(\chi^{-1}(p))$; $f \coloneqq \mathrm{SolveMPP}(\mathcal{G} \restriction T)$
    $x \coloneqq \min(f(T) \cup g(Q))$; $A \coloneqq \mathrm{Attr}_2^{\mathcal{G}}(f^{-1}(x) \cup g^{-1}(x))$
    return $(Q \to \mathbb{R} \cup \{-\infty\} \colon q \mapsto x) \sqcup \mathrm{SolveMPP}(\mathcal{G} \restriction Q \setminus A)$
else
    $T \coloneqq Q \setminus \mathrm{Attr}_2^{\mathcal{G}}(\chi^{-1}(p))$
    if $T = \emptyset$ then return $(Q \to \mathbb{R} \cup \{-\infty\} \colon q \mapsto -\infty)$
    $f \coloneqq \mathrm{SolveMPP}(\mathcal{G} \restriction T)$; $x \coloneqq \max f(T)$; $A \coloneqq \mathrm{Attr}_1^{\mathcal{G}}(f^{-1}(x))$
    return $(Q \to \mathbb{R} \cup \{-\infty\} \colon q \mapsto x) \sqcap \mathrm{SolveMPP}(\mathcal{G} \restriction Q \setminus A)$
end if
```

Lemma 6. *Let $\mathcal{G}$ be a mean-payoff parity game with least priority p odd, $T = Q \setminus \mathrm{Attr}_2(\chi^{-1}(p))$, and $x \in \mathbb{R}$. If $\mathrm{val}^{\mathcal{G}}(q) \geq x$ for some $q \in Q$, then $T \neq \emptyset$ and $\mathrm{val}^{\mathcal{G} \restriction T}(q) \geq x$ for some $q \in T$.*

Theorem 7. *The values of a mean-payoff parity game with d priorities can be computed in time $\mathrm{O}(|Q|^{d+2} \cdot |E| \cdot W)$.*

Proof. We claim that SolveMPP computes, given a mean-payoff parity game $\mathcal{G}$, the function $\mathrm{val}^{\mathcal{G}}$ in the given time bound. Denote by $T(n, m, d)$ the worst-case running time of the algorithm on a game with n states, m edges and d priorities. Note that, if $\mathcal{G}$ has only one priority, then there are no recursive calls to SolveMPP. Since attractors can be computed in time $\mathrm{O}(n + m)$ and the running time of SolveMP is $\mathrm{O}(n^3 \cdot m \cdot W)$, there exists a constant c such that the numbers $T(n, m, d)$ satisfy the following recurrence:

$$\begin{aligned}
&T(1, m, d) \leq c, \\
&T(n, m, 1) \leq c \cdot n^3 \cdot m \cdot W, \\
&T(n, m, d) \leq T(n-1, m, d-1) + T(n-1, m, d) + c \cdot n^3 \cdot m \cdot W\,.
\end{aligned}$$

Solving this recurrence, we get that $T(n, m, d) \leq c \cdot (n+1)^{d+2} \cdot m \cdot W$, which proves the claimed time bound.

It remains to be proved that the algorithm is correct, i.e. that $\mathrm{SolveMPP}(\mathcal{G}) = \mathrm{val}^{\mathcal{G}}$. We prove the claim by induction over the number of states. If there are no states, the claim is trivial. Hence, assume that $Q \neq \emptyset$ and that the claim is true for all games with less than $|Q|$ states. Let $p \coloneqq \min\{\chi(q) \mid q \in Q\}$. We only consider the case that p is even. If p is odd, the proof is similar, but relies on Lemma 6 instead of Lemma 5.

Let T, f, g, x and A be defined as in the corresponding case of the algorithm, and let $f^* = \text{SolveMPP}(\mathcal{G})$. If $\chi(Q) = \{p\}$, then $f^* = g = \text{val}^{(G,0)} = \text{val}^{\mathcal{G}}$, and the claim is fulfilled. Otherwise, by the definition of x and applying the induction hypothesis to the game $\mathcal{G} \upharpoonright T$, we have $\text{val}^{(G,0)}(q) \geq x$ for all $q \in Q$ and $\text{val}^{\mathcal{G}\upharpoonright T}(q) = f(q) \geq x$ for all $q \in T$. Hence, Lemma 5 yields that $\text{val}^{\mathcal{G}}(q) \geq x$ for all $q \in Q$. On the other hand, from any state $q \in A$ Player 2 can play an attractor strategy to $f^{-1}(x) \cup g^{-1}(x)$, followed by an optimal strategy in the game $\mathcal{G} \upharpoonright T$, respectively in the mean-payoff game $(G, 0)$, which ensures that Player 1's payoff does not exceed x. Hence, $\text{val}^{\mathcal{G}}(q) = x = f^*(q)$ for all $q \in A$.

Now, let $q \in Q \setminus A$. We already know that $\text{val}^{\mathcal{G}}(q) \geq x$. Moreover, since $Q \setminus A$ is a 2-trap and applying the induction hypothesis to the game $\mathcal{G} \upharpoonright Q \setminus A$, we have $\text{val}^{\mathcal{G}}(q) \geq \text{val}^{\mathcal{G}\upharpoonright Q\setminus A}(q) = \text{SolveMPP}(\mathcal{G} \upharpoonright Q \setminus A)(q)$. Hence, $\text{val}^{\mathcal{G}}(q) \geq f^*(q)$. To see that $\text{val}^{\mathcal{G}}(q) \leq f^*(q)$, consider the strategy τ of Player 2 that mimics an optimal strategy in $\mathcal{G} \upharpoonright Q \setminus A$ as long as the play stays in $Q \setminus A$ and switches to an optimal strategy in $\mathcal{G}$ as soon as the play reaches A. We have $\text{val}^{\mathcal{G}}(\tau, q) \leq \max\{\text{val}^{\mathcal{G}\upharpoonright Q\setminus A}(q), x\} = f^*(q)$. □

Algorithm SolveMPP is faster and conceptually simpler than the original algorithm proposed for solving mean-payoff parity games [9]. Compared to the recent algorithm proposed by Chatterjee and Doyen [7], which uses a reduction to energy parity games and runs in time $\text{O}(|Q|^{d+4} \cdot |E| \cdot d \cdot W)$, our algorithm has three main advantages: 1. it is faster; 2. it operates directly on mean-payoff parity games, and 3. it is more flexible since it computes the values exactly instead of just comparing them to an integer threshold.

4 Mean-penalty parity games

In this section, we define multi-strategies and *mean-penalty parity games*. We reduce these games to mean-payoff parity games, show that their value problem is in NP ∩ coNP, and propose a deterministic algorithm for computing the values, which runs in pseudo-polynomial time if the number of priorities is bounded.

Syntactically, a *mean-penalty parity game* is a mean-payoff parity game with non-negative weights, i.e. a tuple $\mathcal{G} = (G, \chi)$, where $G = (Q_1, Q_2, E, \text{weight})$ is a weighted game graph with $\text{weight}\colon E \to \mathbb{R}^{\geq 0}$ (or $\text{weight}\colon E \to \mathbb{N}$ for algorithmic purposes), and $\chi\colon Q \to \mathbb{N}$ is a priority function assigning a priority to every state. As for mean-payoff parity games, a play ρ is parity-winning if the minimal priority occurring infinitely often $(\min\{\chi(q) \mid q \in \text{Inf}(\rho)\})$ is even.

Since we are interested in controller synthesis, we define multi-strategies only for Player 1 (who represents the system). Formally, a *multi-strategy* (for Player 1) in $\mathcal{G}$ is a function $\sigma\colon Q^*Q_1 \to \mathcal{P}(Q)\setminus\{\emptyset\}$ such that $\sigma(\gamma q) \subseteq qE$ for all $\gamma \in Q^*$ and $q \in Q_1$. A play ρ of $\mathcal{G}$ is *consistent* with a multi-strategy σ if $\rho(k+1) \in \sigma(\rho[0,k])$ for all $k \in \mathbb{N}$ with $\rho(k) \in Q_1$, and we denote by $\text{Out}^{\mathcal{G}}(\sigma, q_0)$ the set of all plays ρ of $\mathcal{G}$ that are consistent with σ and start in $\rho(0) = q_0$.

Note that, unlike for deterministic strategies, there is, in general, no unique play consistent with a multi-strategy σ for Player 1 and a (deterministic) strat-

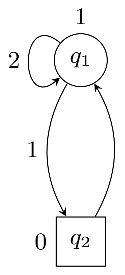

Fig. 2. A mean-penalty parity game

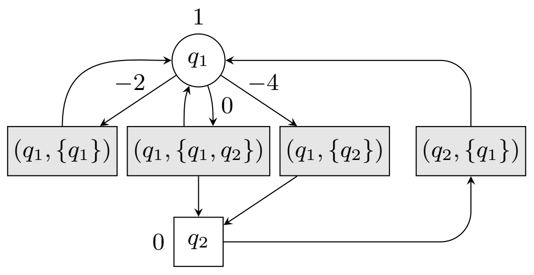

Fig. 3. The corresponding mean-payoff parity game

egy τ for Player 2 from a given initial state. Additionally, note that every deterministic strategy can be viewed as a multi-strategy.

Let $\mathcal{G}$ be a mean-penalty parity game, and let σ be a multi-strategy. We inductively define $\mathrm{penalty}^{\mathcal{G}}_{\sigma}(\gamma)$ (the *total penalty* of γ wrt. σ) for all $\gamma \in Q^*$ by setting $\mathrm{penalty}^{\mathcal{G}}_{\sigma}(\varepsilon) = 0$ as well as $\mathrm{penalty}^{\mathcal{G}}_{\sigma}(\gamma q) = \mathrm{penalty}^{\mathcal{G}}_{\sigma}(\gamma)$ if $q \in Q_2$ and

$$\mathrm{penalty}^{\mathcal{G}}_{\sigma}(\gamma q) = \mathrm{penalty}^{\mathcal{G}}_{\sigma}(\gamma) + \sum_{q' \in qE \setminus \sigma(\gamma q)} \mathrm{weight}(q, q')$$

if $q \in Q_1$. Hence, $\mathrm{penalty}^{\mathcal{G}}_{\sigma}(\gamma)$ is the total weight of transitions blocked by σ along γ. The *mean penalty* of an infinite play ρ is then defined as the average penalty that is incurred along this play in the limit, i.e.

$$\mathrm{penalty}^{\mathcal{G}}_{\sigma}(\rho) = \begin{cases} \limsup\limits_{n\to\infty} \frac{1}{n}\, \mathrm{penalty}^{\mathcal{G}}_{\sigma}(\rho[0,n)) & \text{if } \rho \text{ is parity-winning,} \\ \infty & \text{otherwise.} \end{cases}$$

The mean penalty of a multi-strategy σ from a given initial state q_0 is defined as the supremum over the mean penalties of all plays that are consistent with σ, i.e.

$$\mathrm{penalty}^{\mathcal{G}}(\sigma, q_0) = \sup\{\mathrm{penalty}^{\mathcal{G}}_{\sigma}(\rho) \mid \rho \in \mathrm{Out}^{\mathcal{G}}(\sigma, q_0)\}.$$

The *value* of a state q_0 in a mean-penalty parity game $\mathcal{G}$ is the least mean penalty that a multi-strategy of Player 1 can achieve, i.e. $\mathrm{val}^{\mathcal{G}}(q_0) = \inf_{\sigma} \mathrm{penalty}^{\mathcal{G}}(\sigma, q_0)$, where σ ranges over all multi-strategies of Player 1. A multi-strategy σ is called *optimal* if $\mathrm{penalty}^{\mathcal{G}}(\sigma, q_0) = \mathrm{val}^{\mathcal{G}}(q_0)$ for all $q_0 \in Q$.

Finally, the *value problem for mean-penalty parity games* is the following decision problem: Given a mean-penalty parity game $\mathcal{G} = (G, \chi)$, an initial state $q_0 \in Q$, and a number $x \in \mathbb{Q}$, decide whether $\mathrm{val}^{\mathcal{G}}(q_0) \le x$.

Example 8. Fig. 2 represents a mean-penalty parity game. Note that weights of transitions out of Player 2 states are not indicated as they are irrelevant for the mean penalty. In this game, Player 1 (controlling circle states) has to regularly *block* the self-loop if she wants to enforce infinitely many visits to the state with priority 0. This comes with a penalty of 2. However, the multi-strategy in which she blocks no transition can be played safely for an arbitrary number of times. Hence Player 1 can win with mean-penalty 0 (but infinite memory) by blocking the self-loop once every k moves, where k grows with the number of visits to q_2.

In order to solve mean-penalty games, we reduce them to mean-payoff parity games. We construct from a given mean-penalty parity game $\mathcal{G}$ an exponential-size mean-payoff parity game $\mathcal{G}'$, similar to [3] but with an added priority function. Formally, for a mean-penalty parity game $\mathcal{G} = (G, \chi)$ with game graph $G = (Q_1, Q_2, E, \mathrm{weight})$, the game graph $G' = (Q'_1, Q'_2, E', \mathrm{weight}')$ of the corresponding mean-payoff parity game $\mathcal{G}'$ is defined as follows:

– $Q'_1 = Q_1$ and $Q'_2 = Q_2 \cup \bar{Q}$, where $\bar{Q} := \{(q, F) \mid q \in Q,\ \emptyset \neq F \subseteq qE\}$;
– E' is the (disjoint) union of three kinds of transitions:
  (1) transitions of the form $(q, (q, F))$ for each $q \in Q_1$ and $\emptyset \neq F \subseteq qE$,
  (2) transitions of the form $(q, (q, \{q'\}))$ for each $q \in Q_2$ and $q' \in qE$,
  (3) transitions of the form $((q, F), q')$ for each $q' \in F$;
– the weight function weight' assigns 0 to transitions of type (2) and (3), but $\mathrm{weight}'(q, (q, F)) = -2 \sum_{q' \in qE \setminus F} \mathrm{weight}(q, q')$ to transitions of type (1).

Finally, the priority function χ' of $\mathcal{G}'$ coincides with χ on Q and assigns priority $M := \max\{\chi(q) \mid q \in Q\}$ to all states in $\bar{Q}$.

Example 9. Fig. 3 depicts the mean-payoff parity game obtained from the mean-penalty parity game from Example 8, depicted in Fig. 2.

The correspondence between $\mathcal{G}$ and $\mathcal{G}'$ is expressed in the following lemma.

Lemma 10. *Let $\mathcal{G}$ be a mean-penalty parity game, $\mathcal{G}'$ the corresponding mean-payoff parity game, and $q_0 \in Q$.*

1. *For every multi-strategy σ in $\mathcal{G}$ there exists a strategy σ' for Player 1 in $\mathcal{G}'$ such that* $\mathrm{val}(\sigma', q_0) \geq -\mathrm{penalty}(\sigma, q_0)$.
2. *For every strategy σ' for Player 1 in $\mathcal{G}'$ there exists a multi-strategy σ in $\mathcal{G}$ such that* $\mathrm{penalty}(\sigma, q_0) \leq -\mathrm{val}(\sigma', q_0)$.
3. $\mathrm{val}^{\mathcal{G}'}(q_0) = -\mathrm{val}^{\mathcal{G}}(q_0)$.

It follows from Theorem 2 and Lemma 10 that every mean-penalty parity game admits an optimal multi-strategy.

Corollary 11. *In every mean-penalty parity game, Player 1 has an optimal multi-strategy.*

We now show that Player 2 has a memoryless optimal strategy of a special kind in the mean-payoff parity game derived from a mean-penalty parity game. This puts the value problem for mean-penalty parity games into coNP, and is also a crucial point in the proof of Lemma 14 below.

Lemma 12. *Let $\mathcal{G}$ be a mean-penalty parity game and $\mathcal{G}'$ the corresponding mean-payoff parity game. Then in $\mathcal{G}'$ there is a memoryless optimal strategy τ' for Player 2 such that for every $q \in Q$ there exists a total order $\leq_q$ on the set qE with $\tau'((q, F)) = \min_{\leq_q} F$ for every state $(q, F) \in \bar{Q}$.*

Proof (Sketch). Let τ be a memoryless optimal strategy for Player 2 in $\mathcal{G}'$. For a state q, we consider the set qE and order it in the following way. We inductively define $F_1 = qE$, $q_i = \tau((q, F_i))$ and $F_{i+1} = F_i \setminus \{q_i\}$ for every $1 \leq i \leq |qE|$. Note that $\{q_1, \dots, q_{|qE|}\} = qE$. We set $q_1 \leq_q q_2 \leq_q \dots \leq_q q_{|qE|}$ and define a new memoryless strategy τ' for Player 2 in $\mathcal{G}'$ by $\tau'((q, F)) = \min_{\leq_q} F$ for $(q, F) \in \bar{Q}$ and $\tau'(q) = \tau(q)$ for all $q \in Q_2$. It can be shown that $\text{val}(\tau', q_0) \leq \text{val}(\tau, q_0)$ for all $q_0 \in Q$, which proves that τ' is optimal. $\square$

In order to put the value problem for mean-penalty parity games into NP $\cap$ coNP, we propose a more sophisticated reduction from mean-penalty parity games to mean-payoff parity games, which results in a polynomial-size mean-payoff parity game. Intuitively, in a state $q \in Q_1$ we ask Player 1 *consecutively* for each outgoing transition whether he wants to block that transition. If he allows a transition, then Player 2 has to decide whether she wishes to explore this transition. Finally, after all transitions have been processed in this way, the play proceeds along the *last* transition that Player 2 has desired to explore.

Formally, let us fix a mean-penalty parity game $\mathcal{G} = (G, \chi)$ with game graph $G = (Q_1, Q_2, E, \text{weight})$, and denote by $k \coloneqq \max\{|qE| \mid q \in Q\}$ the maximal out-degree of a state. Then the polynomial-size mean-payoff parity game $\mathcal{G}''$ has vertices of the form q and (q, a, i, m), where $q \in Q$, $a \in \{\text{choose}, \text{allow}, \text{block}\}$, $i \in \{1, \dots, k+1\}$ and $m \in \{0, \dots, k\}$; vertices of the form q and (q, choose, i, m) belong to Player 1, while vertices of the form (q, allow, i, m) or (q, block, i, m) belong to Player 2. To describe the transition structure of $\mathcal{G}$, let $q \in Q$ and assume that $qE = \{q_1, \dots, q_k\}$ (a state may occur more than once in this list). Then the following transitions originate in a state of the form q or (q, a, i, m):

1. a transition from q to $(q, \text{choose}, 1, 0)$ with weight 0,
2. for all $1 \leq i \leq k$ and $0 \leq m \leq k$ a transition from (q, choose, i, m) to (q, allow, i, m) with weight 0,
3. if $q \in Q_1$ then for all $1 \leq i \leq k$ and $0 \leq m \leq k$ a transition from (q, choose, i, m) to (q, block, i, m) with weight 0, *except* if $i = k$ and $m = 0$;
4. for all $0 \leq m \leq k$ a transition from $(q, \text{choose}, k+1, m)$ to q_m with weight 0 (where q_0 can be chosen arbitrarily),
5. for all $1 \leq i \leq k$ and $0 \leq m \leq k$ a transition from (q, allow, i, m) to $(q, \text{choose}, i+1, i)$ with weight 0,
6. for all $1 \leq i \leq k$ and $1 \leq m \leq k$ a transition from (q, allow, i, m) to $(q, \text{choose}, i+1, m)$ with weight 0,
7. for all $1 \leq i \leq k$ and $0 \leq m \leq k$ a transition from (q, block, i, m) to $(q, \text{choose}, i+1, m)$ with weight $-2(k+1) \cdot \text{weight}(q, q_i)$.

Finally, the priority of a state $q \in Q$ equals the priority of the same state in $\mathcal{G}$, whereas all states of the form (q, a, i, m) have priority $M = \max\{\chi(q) \mid q \in Q\}$.

Example 13. For the game of Fig. 2, this transformation would yield the game depicted in Fig. 4. In this picture, a, b and c stand for *allow*, *block* and *choose*, respectively; zero weights are omitted.

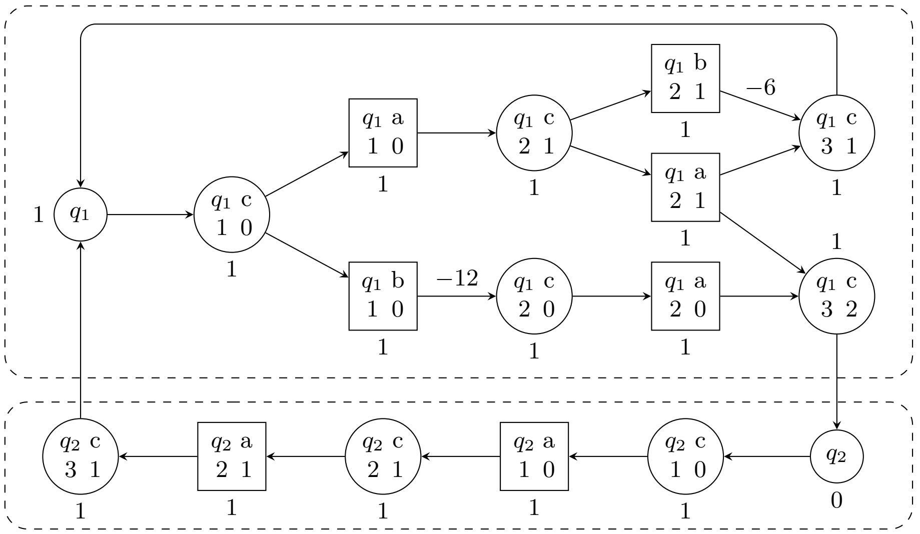

Fig. 4. The game $\mathcal{G}''$ associated with the game $\mathcal{G}$ of Fig. 2

It is easy to see that the game $\mathcal{G}''$ has polynomial size and can, in fact, be constructed in polynomial time from the given mean-penalty parity game $\mathcal{G}$. The following lemma relates the game $\mathcal{G}''$ to the mean-payoff parity game $\mathcal{G}'$ of exponential size constructed earlier and to the original game $\mathcal{G}$.

Lemma 14. *Let $\mathcal{G}$ be a mean-penalty parity game, $\mathcal{G}'$ the corresponding mean-payoff parity game of exponential size, $\mathcal{G}''$ the corresponding mean-payoff parity game of polynomial size, and $q_0 \in Q$.*

1. *For every multi strategy σ in $\mathcal{G}$ there exists a strategy σ' for Player 1 in $\mathcal{G}''$ such that* $\mathrm{val}(\sigma', q_0) \geq -\operatorname{penalty}(\sigma, q_0)$.
2. *For every strategy τ for Player 2 in $\mathcal{G}'$ there exists a strategy τ' for Player 2 in $\mathcal{G}''$ such that* $\mathrm{val}(\tau', q_0) \leq \mathrm{val}(\tau, q_0)$.
3. $\mathrm{val}^{\mathcal{G}''}(q_0) = -\mathrm{val}^{\mathcal{G}}(q_0)$.

Since the mean-payoff game $\mathcal{G}''$ can be computed from $\mathcal{G}$ in polynomial time, we obtain a polynomial-time many-one reduction from the value problem for mean-penalty parity games to the value problem for mean-payoff parity games. By Corollary 3 and Theorem 4, the latter problem belongs to NP ∩ coNP.

Theorem 15. *The value problem for mean-penalty parity games belongs to NP ∩ coNP.*

A deterministic algorithm. Naturally, we can use the polynomial translation from mean-penalty parity games to mean-payoff parity games to solve mean-penalty parity games deterministically. Note that the mean-payoff parity game $\mathcal{G}''$ derived from a mean-penalty parity game has $\mathrm{O}(|Q| \cdot k^2)$ states and $\mathrm{O}(|Q| \cdot k^2)$

Algorithm SymbSolveMPP($\mathcal{G}$)

```
Input: mean-penalty parity game $\mathcal{G} = (G, \chi)$
Output: $\mathrm{val}^{\mathcal{G}}$

if $Q = \emptyset$ then return $\emptyset$
$p \coloneqq \min\{\chi(q) \mid q \in Q\}$
if $p$ is even then
   $g \coloneqq \mathrm{SymbSolveMP}(G, 0)$
   if $\chi(q) = p$ for all $q \in Q$ then return $g$
   $T \coloneqq Q \setminus \mathrm{Attr}_1^{\mathcal{G}}(\chi^{-1}(p))$; $f \coloneqq \mathrm{SymbSolveMPP}(\mathcal{G} \restriction T)$
   $x \coloneqq \max(f(T) \cup g(Q))$; $A \coloneqq \mathrm{Attr}_2^{\mathcal{G}}(f^{-1}(x) \cup g^{-1}(x))$
   return $(Q \to \mathbb{R} \cup \{\infty\} \colon q \mapsto x) \sqcap \mathrm{SymbSolveMPP}(\mathcal{G} \restriction Q \setminus A)$
else
   $T \coloneqq Q \setminus \mathrm{Attr}_2^{\mathcal{G}}(\chi^{-1}(p))$
   if $T = \emptyset$ then return $(Q \to \mathbb{R} \cup \{\infty\} \colon q \mapsto \infty)$
   $f \coloneqq \mathrm{SymbSolveMPP}(\mathcal{G} \restriction T)$; $x \coloneqq \min f(T)$; $A \coloneqq \mathrm{Attr}_1^{\mathcal{G}}(f^{-1}(x))$
   return $(Q \to \mathbb{R} \cup \{\infty\} \colon q \mapsto x) \sqcup \mathrm{SymbSolveMPP}(\mathcal{G} \restriction Q \setminus A)$
end if
```

edges, where k is the maximum out-degree of a state in $\mathcal{G}$; the number of priorities remains constant. Moreover, if weights are given in integers and W is the highest absolute weight in $\mathcal{G}$, then the highest absolute weight in $\mathcal{G}''$ is $\mathrm{O}(k \cdot W)$. Using Theorem 7, we thus obtain a deterministic algorithm for solving mean-penalty parity games that runs in time $\mathrm{O}(|Q|^{d+3} \cdot k^{2d+7} \cdot W)$. If k is a constant, the running time is $\mathrm{O}(|Q|^{d+3} \cdot W)$, which is acceptable. In the general case however, the best upper bound on k is the number of states, and we get an algorithm that runs in time $\mathrm{O}(|Q|^{3d+10} \cdot W)$. Even if the numbers of priorities is small, this running time would not be acceptable in practical applications.

The goal of this section is to show that we can do better; namely we will give an algorithm that runs in time $\mathrm{O}(|Q|^{d+3} \cdot |E| \cdot W)$, independently of the maximum out-degree. The idea is as follows: we use Algorithm SolveMPP on the mean-payoff parity game $\mathcal{G}'$ of exponential size, but we show that we can run it *on* $\mathcal{G}$, i.e., by handling the extra states of $\mathcal{G}'$ symbolically during the computation. As a first step, we adapt the pseudo-polynomial algorithm by Zwick and Paterson [22] to compute the values of a mean-penalty parity game with a trivial parity objective.

Lemma 16. *The values of a mean-penalty parity game with priority function* $\chi \equiv 0$ *can be computed in time* $\mathrm{O}(|Q|^4 \cdot |E| \cdot W)$.

Algorithm SymbSolveMPP is our algorithm for computing the values of a mean-penalty parity game. The algorithm employs as a subroutine an algorithm SymbSolveMP for computing the values of a mean-penalty parity with a trivial priority function (see Lemma 16). Since SymbSolveMP can be implemented to run in time $\mathrm{O}(|Q|^4 \cdot |E| \cdot W)$, the running time of the procedure SymbSolveMPP is $\mathrm{O}(|Q|^{d+3} \cdot |E| \cdot W)$. Notably, the algorithm runs in polynomial time if the number of priorities is bounded and we are only interested in the average *number* of edges blocked by a strategy in each step (i.e. if all weights are equal to 1).

Theorem 17. *The values of a mean-penalty parity game with d priorities can be computed in time* $\mathrm{O}(|Q|^{d+3} \cdot |E| \cdot W)$.

Proof (Sketch). From Lemma 16 and with the same runtime analysis as in the proof of Theorem 7, we get that SymbSolveMPP runs in time $\mathrm{O}(|Q|^{d+3} \cdot |E| \cdot W)$. To prove that the algorithm is correct, we show that there is a correspondence between the values the algorithm computes on a mean-penalty parity game $\mathcal{G}$ and the values computed by Algorithm SolveMPP on the mean-payoff parity game $\mathcal{G}'$. More precisely, we show that $\mathrm{SolveMPP}(\mathcal{G}') \upharpoonright Q = -\mathrm{SymbSolveMPP}(\mathcal{G})$. The correctness of the algorithm thus follows from Lemma 10, which states that $\mathrm{val}^{\mathcal{G}'} \upharpoonright Q = -\mathrm{val}^{\mathcal{G}}$. □

5 Conclusion

In this paper, we have studied mean-payoff parity games, with an application to finding permissive strategies in parity games with penalties. In particular, we have established that mean-penalty parity games are not harder to solve than mean-payoff parity games: for both kinds of games, the value problem is in $\mathrm{NP} \cap \mathrm{coNP}$ and can be solved by an exponential algorithm that becomes pseudo-polynomial when the number of priorities is bounded.

One complication with both kinds of games is that optimal strategies for Player 1 require infinite memory, which makes it hard to synthesise these strategies. A suitable alternative to optimal strategies are ε*-optimal* strategies that achieve the value of the game by at most ε. Since finite-memory ε-optimal strategies are guaranteed to exist [2], a challenge for future work is to modify our algorithms so that they compute not only the values of the game but also a finite-memory ε-optimal (multi-)strategy for Player 1.

Acknowledgement. We thank an anonymous reviewer for pointing out the polynomial reduction from mean-penalty parity games to mean-payoff parity games, which has simplified the proof that mean-penalty parity games are in NP.

References

1. J. Bernet, D. Janin, I. Walukiewicz. Permissive strategies: from parity games to safety games. *RAIRO – ITA*, 36(3):261–275, 2002.
2. R. Bloem, K. Chatterjee, T. A. Henzinger, B. Jobstmann. Better quality in synthesis through quantitative objectives. In *CAV'09*, vol. 5643 of *LNCS*, pp. 140–156. Springer-Verlag, 2009.
3. P. Bouyer, M. Duflot, N. Markey, G. Renault. Measuring permissivity in finite games. In *CONCUR'09*, vol. 5710 of *LNCS*, pp. 196–210. Springer-Verlag, 2009.

4. P. Bouyer, U. Fahrenberg, K. G. Larsen, N. Markey, J. Srba. Infinite runs in weighted timed automata with energy constraints. In *FORMATS'08*, vol. 5215 of *LNCS*, pp. 33–47. Springer-Verlag, 2008.
5. P. Bouyer, N. Markey, J. Olschewski, M. Ummels. Measuring permissiveness in parity games: Mean-payoff parity games revisited. Research Report LSV-11-17, Laboratoire Spécification et Vérification, ENS Cachan, France, 2011.
6. A. Chakrabarti, L. de Alfaro, T. A. Henzinger, M. Stoelinga. Resource interfaces. In *EMSOFT'03*, vol. 2855 of *LNCS*, pp. 117–133. Springer-Verlag, 2003.
7. K. Chatterjee, L. Doyen. Energy parity games. In *ICALP'10 (2)*, vol. 6199 of *LNCS*, pp. 599–610. Springer-Verlag, 2010.
8. K. Chatterjee, L. Doyen, T. A. Henzinger, J.-F. Raskin. Generalized mean-payoff and energy games. In *FSTTCS'10*, vol. 8 of *LIPIcs*, pp. 505–516. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2010.
9. K. Chatterjee, T. A. Henzinger, M. Jurdziński. Mean-payoff parity games. In *LICS'05*, pp. 178–187. IEEE Computer Society Press, 2005.
10. K. Chatterjee, T. A. Henzinger, N. Piterman. Generalized parity games. In *FoSSaCS'07*, vol. 4423 of *LNCS*, pp. 153–167. Springer-Verlag, 2007.
11. A. Ehrenfeucht, J. Mycielski. Positional strategies for mean payoff games. *Int. Journal of Game Theory*, 8(2):109–113, 1979.
12. E. A. Emerson, C. S. Jutla. Tree automata, mu-calculus and determinacy. In *FOCS'91*, pp. 368–377. IEEE Computer Society Press, 1991.
13. M. Jurdziński. Deciding the winner in parity games is in UP $\cap$ co-UP. *Information Processing Letters*, 68(3):119–124, 1998.
14. R. M. Karp. A characterization of the minimum cycle mean in a digraph. *Discrete Mathematics*, 23(3):309–311, 1978.
15. E. Kopczyński. Half-positional determinacy of infinite games. In *ICALP 2006 (2)*, vol. 4052 of *LNCS*, pp. 336–347. Springer-Verlag, 2006.
16. M. Luttenberger. Strategy iteration using non-deterministic strategies for solving parity games. Research Report cs.GT/0806.2923, arXiv, 2008.
17. D. A. Martin. Borel determinacy. *Annals of Mathematics*, 102:363–371, 1975.
18. A. W. Mostowski. Games with forbidden positions. Tech. Rep. 78, Instytut Matematyki, Uniwersytet Gdański, Poland, 1991.
19. S. Pinchinat, S. Riedweg. You can always compute maximally permissive controllers under partial observation when they exist. In *ACC'05*, pp. 2287–2292. 2005.
20. W. Thomas. Infinite games and verification (extended abstract of a tutorial). In *CAV'04*, vol. 2404 of *LNCS*, pp. 58–64. Springer-Verlag, 2002.
21. W. Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theoretical Computer Science*, 200(1–2):135–183, 1998.
22. U. Zwick, M. Paterson. The complexity of mean payoff games on graphs. *Theoretical Computer Science*, 158(1&2):343–359, 1996.