Fast value iteration: A uniform approach to efficient algorithms for energy games

Michaël Cadilhac¹, Antonio Casares², and Pierre Ohlmann³
¹ DePaul University, Chicago, IL, USA
² University of Warsaw, Poland
³ CNRS, Université Aix-Marseille, LIS, France

Abstract. We study algorithms for solving parity, mean-payoff and energy games. We propose a systematic framework, which we call Fast value iteration, for describing, comparing, and proving correctness of such algorithms. The approach is based on potential reductions, as introduced by Gurvich, Karzanov and Khachiyan (1988). This framework allows us to provide simple presentations and correctness proofs of known algorithms, unifying the Optimal strategy improvement algorithm by Schewe (2008) and the quasi dominions approach by Benerecetti et al. (2020), amongst others. The new approach also leads to novel symmetric versions of these algorithms, highly efficient in practice, but for which we are unable to prove termination. We report on empirical evaluation, comparing the different fast value iteration algorithms, and showing that they are competitive even to top parity game solvers.

Keywords: Mean-payoff games · energy games · pseudopolynomial algorithm · value iteration

1 Introduction

Mean-payoff and energy games. The games under study are infinite duration games where two players, Min and Max, move a token over a finite directed graph with no sink, where the edges of the graph are labelled by payoffs in Z. When playing a mean-payoff game, the players optimise (minimise or maximise, respectively) the asymptotic average payoff. In an energy game, they instead optimise the supremum cumulative sum of payoffs within [0, ∞]. These games are positionaly determined [12,5]; the two players can play optimally even when restricted to strategies that only depend on the current position of the game. We refer to Figure 1 for a complete example.

In this paper, we are interested in the problem of computing energy values of the vertices in a given game which we call solving the energy game. It easily follows from positional determinacy that the energy value of a vertex is finite if and only if its mean-payoff value is non-positive [6]. Therefore solving an energy game also solves the so called threshold problem for the associated mean-payoff game. As it turns out, all state-of-the-art algorithms [2,4,6,11,31,33] for the mean-payoff threshold problem actually solve the energy game.
Fig. 1. Example of a game; circles belong to Min and squares belong to Max. From left to right, the mean-payoff values are $-2$, $-2$, $-\frac{1}{4}$, $-\frac{1}{4}$, $1$ and $1$, and positional strategies for mean-payoff values are identified in bold. Energy values are $0$, $2$, $9$, $0$, $\infty$ and $\infty$, and with optimal strategies given by the double-headed arrows.

Mean-payoff values achieved by positional strategies can be computed in polynomial time, and therefore the threshold problem belongs to $\text{NP} \cap \text{coNP}$. Despite numerous efforts, no polynomial algorithm is known. Mean-payoff games are known [32] to generalise parity games [13,30] which also belong to $\text{NP} \cap \text{coNP}$ but for which algorithms with quasipolynomial runtime were recently devised [7]. However, quasipolynomial algorithms for parity games do not generalise to mean-payoff games [15].

Algorithmic paradigms. There are two well-established paradigms for solving energy games: value iteration (sometimes called “progress measure”) and strategy improvement. The standard value iteration for energy games (which we will call Simple value iteration, SVI for short) was introduced by Brim et al. [6]. While subject to good theoretical (pseudopolynomial) bounds, it is well-known to be prohibitively slow in practice, as its worst-case behaviour is frequently displayed. On the other hand, strategy improvement algorithms [3] typically solve practical instances in a constant number of iterations. Although it offers a useful categorization of older algorithms, the value iteration versus strategy improvement dichotomy fails to accurately describe a new wave of efficient algorithms.

In recent years, multiple hybrid algorithms – borrowing ideas from both paradigms – have been put forward. In 2008, Schewe [33] introduced an algorithm called Optimal strategy improvement (OSI) for solving parity or mean-payoff games. As explained by Luttenberger [27], Schewe’s presentation of OSI is in fact closer to value iteration, but it can also be formally cast as a strategy improvement in a carefully generalised framework allowing for nondeterministic strategies. In 2019, Dorfman et al. [11] presented a value iteration method augmented by a carefully crafted acceleration mechanism (which we call DKZ), thereby improving on the best theoretical guarantees (this algorithm can be seen as a reformulation of the GKK algorithm [19], see also [31] for further analyses). Based on the idea of quasi dominions (similar to Fearnley’s snares [14] in a strategy improvement context), Benerecetti et al. [2] proposed another such acceleration mechanism, obtaining
Contributions. Our contributions are as follows.

(1) Fast value iteration framework. We consider potential reductions, as introduced by Gurvich, Karzanov and Khachiyan [19], to design a systematic method for producing algorithms for energy games, which we call the fast value iteration framework. A potential is a mapping which assigns a positive weight to each vertex. Such a potential naturally induces a transformation (a potential reduction) of the game, which preserves the weight of every cycle and thus the values in the mean-payoff game. The fast value iteration meta-algorithm (Algorithm 1) simply iterates on potential reductions until a fixpoint is reached. This meta-algorithm can be instantiated on any given class of potentials, leading to different algorithms, whose correctness is automatically guaranteed under mild assumptions on the potentials (Theorem 4). Interestingly, the framework also provides a symmetric meta-algorithm, for which termination is observed in practice, but we have not been able to prove it theoretically.

The algorithms from the fast value iteration framework share some properties that make them convenient for practical applications. The main reason STRIX uses OSI is its support for modularity. Since games coming from LTL-formulas are typically huge, an important feature is to be able to solve them piecewise, avoiding loading the entire game into memory. We show that all algorithms within the fast value iteration framework are well-suited for this modular approach, which also opens exciting perspectives for parallelised implementations.

(2) Unifying and simplifying existing algorithms. We revisit various algorithms in the light of the above framework. Naturally, the classic SVI [6] is captured (Example 6), as well as the algorithms GKK [19] and DKZ [11] (Section 4.3), whose original presentations fit the potential reduction framework.

More interestingly, we also capture algorithms showcasing an excellent performance in practice, defying the common belief that VI algorithms are slow. We unify and simplify the algorithms OSI by Schewe [33] and the involved QDPM by Benerecetti et al. [2]. Our presentations are streamlined (see Section 4 for details), leading to immediate correctness proofs. It also allows to isolate the core algorithmic idea underlying these two algorithms, which is a natural adaptation of Dijkstra’s algorithm to the two-player setting. We call the obtained reinterpretation of OSI and QDPM within the fast value iteration framework, the Positive Path Iteration (PPI).
The abstraction provided by our approach sets the stage to easily craft new algorithms. Showcasing its applicability, we propose a dynamic version of PPI (DPPI), which provably breaks the theoretical barrier set by OSI and QDPM (Lemma 13). Many possibilities for future work are proposed in the conclusion.

(3) Empirical evaluation. We compare the implementations of the algorithms described in the fast value iteration framework to OSI and QDPM, as well as to the top parity game solvers. This evaluation shows: (i) fast value iteration algorithms are highly efficient in practice, and especially robust towards hard instances; (ii) alternating versions of the algorithms not only terminate, but are remarkably efficient.

2 Preliminaries

A game is a tuple \( G = (G, w, V_{\text{Min}}, V_{\text{Max}}) \), where \( G = (V, E) \) is a finite sinkless directed graph, \( w : E \rightarrow \mathbb{Z} \) is a labelling of its edges by integer weights, and \( V_{\text{Min}}, V_{\text{Max}} \) is a partition of \( V \). We set \( n = |V|, m = |E| \) and \( W = \max_{e \in E} |w(e)| \).

We say that vertices in \( V_{\text{Min}} \) belong to Min and that those in \( V_{\text{Max}} \) belong to Max. We now fix a game \( G = (G, w, V_{\text{Min}}, V_{\text{Max}}) \).

We simply write \( vv' \) for an edge \((v, v') \in E\). A path is a (possibly empty, possibly infinite) sequence of edges \( \pi = e_0e_1 \ldots \), with \( e_i = v_iv'_i \), such that \( v'_i = v_{i+1} \). We write \( v_0 \rightarrow v_1 \rightarrow \ldots \) to denote such a path. The sum of a finite path \( \pi \) is the sum of the weights appearing on it, we denote it by \( \text{sum}(\pi) \). Given a finite or infinite path \( \pi = e_0e_1 \ldots \) and an integer \( k \geq 0 \), we let \( \pi_{\leq k} = e_0e_1 \ldots e_{k-1} \), and we let \( \pi_{< k} = \pi_{< k+1} \). Note that \( \pi_{< 0} \) is the empty path, and that \( \pi_{< k} \) has length \( k \). By convention, the empty path starts and ends in all vertices.

A valuation is a map \( \text{val} : \mathbb{Z}^\omega \rightarrow \mathbb{R} \cup \{\infty\} \) assigning a potentially infinite value to infinite sequences of weights. We use \( \mathbb{R}^\infty, \mathbb{Z}^\infty \) and \( \mathbb{N}^\infty \) to denote respectively \( \mathbb{R} \cup \{\infty\}, \mathbb{Z} \cup \{\infty\} \) and \( \mathbb{N} \cup \{\infty\} \). The four valuations studied in this paper are the mean-payoff, energy, positive-energy, and first-if-positive valuations given by:

\[
\text{MP}(w) = \limsup_k \frac{1}{k} \sum_{i=0}^{k-1} w_i \in \mathbb{R}, \quad \text{En}^+(w) = \sum_{i=0}^{k_{\text{neg}}-1} w_i \in \mathbb{N}^\infty, \\
\text{En}(w) = \sup_k \sum_{i=0}^{k-1} w_i \in \mathbb{N}^\infty, \quad \text{First}^+(w) = \max(w_0, 0) \in \mathbb{N},
\]

where \( w = w_0w_1 \ldots \) is a sequence of weights and \( k_{\text{neg}} = \min\{k \mid w_k < 0\} \in \mathbb{N}^\infty \) is the first index of a negative weight. For technical convenience, we will also consider games in which weights are potentially (positively) infinite. We extend the definitions of \( \text{En}, \text{En}^+ \) and \( \text{First}^+ \) to words in \((\mathbb{Z}^\infty)^\omega\), using the same formula. Note that for any \( w \in (\mathbb{Z}^\infty)^\omega \) we have \( \text{En}^+ \leq \text{En} \). The four valuations are illustrated on a given sequence of weights in Figure 2.

A strategy for Min is a map \( \sigma : V_{\text{Min}} \rightarrow E \) such that for all \( v \in V_{\text{Min}} \), it holds that \( \sigma(v) \) is an edge outgoing from \( v \). We say that a (finite or infinite) path \( \pi = e_0e_1 \ldots \) is consistent with \( \sigma \) if whenever \( e_i = v_iv_{i+1} \) is defined and \( v_i \in V_{\text{Min}} \), it holds that \( e_i = \sigma(v_i) \). We write in this case \( \pi \models \sigma \). Strategies for Max are defined similarly and written \( \tau : V_{\text{Max}} \rightarrow E \). The theorem below states
that the three valuations are determined with positional strategies. It is well known for MP and En and easy to prove for En+. We remark that positional determinacy also holds for the two energy valuations En and En+ over games where we allow for infinite weights.

**Theorem 1 ([12,5]).** For each \( \text{val} \in \{ \text{MP}, \text{En}, \text{En}^+ \} \), there exist strategies \( \sigma_0 \) for Min and \( \tau_0 \) for Max such that for all \( v \in V \) we have

\[
\sup_{\pi|\sigma_0} \text{val}(w(\pi)) = \inf_{\sigma} \sup_{\pi|\sigma} \text{val}(w(\pi)) = \sup_{\pi|\tau} \inf_{\tau} \text{val}(w(\pi)) = \inf_{\pi|\tau_0} \text{val}(w(\pi)),
\]

where \( \sigma, \tau \) and \( \pi \) respectively range over strategies for Min, strategies for Max, and infinite paths from \( v \).

The quantity defined by the equilibrium above is called the **value** of \( v \) in the val game, and we denote it by \( \text{val}_G(v) \in \mathbb{R}^\infty \); the strategies \( \sigma_0 \) and \( \tau_0 \) are called **val-optimal**, note that they do not depend on \( v \). The two main algorithmic problems we are interested in are (i) computing the value \( \text{En}_G(v) \) of a given vertex \( v \) in a game, and (ii) decide whether \( \text{MP}_G(v) \leq 0 \) (threshold problem). The following result relates the values in the mean-payoff and energy games; this direct consequence of Theorem 1 was first stated in [6].

**Corollary 2 ([6]).** For all \( v \in V \) it holds that

\[
\text{MP}_G(v) \leq 0 \iff \text{En}_G(v) < \infty \iff \text{En}_G(v) \leq (n - 1)W.
\]

Therefore computing En-values of the games is harder than the mean-payoff threshold problem. It is easy to deduce En-optimal strategies for Min from the knowledge of the En-values: we select Min-edges that minimise the sum of the edge’s weight and the energy of the destination. However no knowledge is gained about Max strategies besides the winning region (over which En values are \( \infty \)). As explained in the introduction, all state-of-the-art algorithms for the threshold problem actually compute En values. This shifts our focus from mean-payoff to energy games.
Attractors. Given a subset \( S \subseteq V \), the attractor \( \text{Attr}^\text{Max}_G(S) \) to \( S \) in \( G \) is defined to be the set of vertices \( v \) such that Max can ensure to reach \( S \) from \( v \).

Simple games. A finite path \( v_0 \rightarrow v_1 \rightarrow \ldots \rightarrow v_k \) is simple if there is no repetition in \( v_0, v_1, \ldots, v_{k-1} \); note that a cycle may be simple. A game is simple if all simple cycles have nonzero sum. The following result is folklore and states that one may reduce to a simple game at the cost of a linear blow up on \( W \).

It holds thanks to the fact that positive mean-payoff values are \( \geq 1/n \) (with \( n = |V| \)), which is a well-known consequence of Theorem 1.

Lemma 3. Let \( G = (G, w, V_{\text{Min}}, V_{\text{Max}}) \) be an arbitrary game. The game \( G' = (G, w', V_{\text{Min}}, V_{\text{Max}}) \), with \( w' = (n+1)w - 1 \), is simple and has the same vertices of positive mean-payoff values as \( G \).

3 Fast value iteration: A meta-algorithm based on potential reductions

3.1 Potential reductions

Fix a game \( G = (G = (V, E), w, V_{\text{Min}}, V_{\text{Max}}) \). A potential is a map \( \varphi : V \rightarrow \mathbb{N}^\infty \).

Potentials are partially ordered coordinatewise. We write \( \varphi = 0 \) if \( \varphi(v) = 0 \) for all \( v \in V \). Given an edge \( vv' \in E \), we define its \( \varphi \)-modified weight to be

\[
 w_\varphi(vv') = \begin{cases} 
 \infty & \text{if } \varphi(v), \varphi(v') \text{ or } w(vv') \text{ is } \infty, \\
 w(vv') + \varphi(v') - \varphi(v) & \text{otherwise.}
\end{cases}
\]

The \( \varphi \)-modified game \( G_\varphi \) is simply the game \( (G, w_\varphi, V_{\text{Min}}, V_{\text{Max}}) \); informally, all weights are replaced by the modified weights. Note that the underlying graph does not change, in particular paths in \( G \) and \( G_\varphi \) are the same. Moving from \( G \) to \( G_\varphi \) for a given potential \( \varphi \) is called a potential reduction.

Weights of cycles are preserved by finite potential reductions, and therefore, as an easy consequence of positionality (Theorem 1), mean-payoff values are preserved. Note that any edge outgoing from a vertex \( v \) with potential \( \varphi(v) = \infty \) has weight \( \infty \) in the modified game, therefore \( v \) has En and En\(^+\)-values \( \infty \) in \( G_\varphi \). Note also that sequential applications of potential reductions correspond to reducing with respect to the sum of the potentials: \( (G_\varphi)_\varphi' = G_{\varphi + \varphi'} \).

Potential reductions were introduced by Gallai [18] for studying network-related problems such as shortest-paths problems. In the context of mean-payoff or energy games, they were introduced in [19] and later sometimes rediscovered.

3.2 The fast value iteration meta-algorithm

A potential assigner is a function \( \Psi \) that assigns a potential \( \Psi(G) : V \rightarrow \mathbb{N}^\infty \) to each game \( G \). A potential assigner \( \Psi \) induces a fast value iteration algorithm (called \( \Psi \)-FVI) as follows: successively apply potential reductions using the potentials given by \( \Psi \), until a game \( G' \) is reached with \( \Psi(G')(V) \subseteq \{0, \infty\} \). For
an arbitrary potential assigner, this algorithm might not terminate, or provide a
final game $G'$ carrying irrelevant information. However, we show that under mild
hypotheses on $\Psi$, this algorithm terminates, and $\text{En}_{G'} = \Psi(G')$, with the vertices
with $\text{En}$-value 0 corresponding to the vertices with finite value in the original
game. Moreover, the exact $\text{En}$-values of the original game can be recovered from
the sequence of potentials obtained during the computation.

We formalise this idea in Algorithm 1 and Theorem 4. To ensure termination,
we need to artificially increase the potential of some vertices to $\infty$ whenever a
threshold is reached, and then remove Max's attractor to $\infty$. This technique is
standard in value iteration algorithms, see e.g. [6].

Algorithm 1 $\Psi$-Fast value iteration algorithm.

| Step | Description |
|------|-------------|
| 1    | $\Phi \leftarrow 0$ $\triangleright \Phi$ carries the cumulative sum of potentials |
| 2    | do |
| 3    | $\varphi \leftarrow \Psi(G)$ |
| 4    | $\Phi \leftarrow \Phi + \varphi$ $\triangleright$ Update cumulative sum over $G$ |
| 5    | $G \leftarrow G^\varphi$ |
| 6    | $A \leftarrow \text{Attr}_{\max}^G((n-1)W + 1, \infty))$ |
| 7    | Set $\Phi(v) = \infty$ for all $v \in A$ |
| 8    | $G \leftarrow G \setminus A$ |
| 9    | while $\varphi \neq 0$ and $G \neq \emptyset$ |
| 10   | return $\Phi$ |

Let us isolate two relevant properties of potential assigners: (1) **Soundness**:
for any game $G$, $\Psi(G) \leq \text{En}_G$; (2) **Completeness**: for any $G$, if $\Psi(G) = 0$ then $\text{En}_G = 0$. We also say that a potential $\varphi$ is sound over a given game if condition
(1) is met. We may now state our first main result.

**Theorem 4.** Let $\Psi$ be a sound and complete potential assigner. Then Algo-

rithm 1 terminates in at most $n^2W$ iterations, and returns $\Phi = \text{En}_G$.

**Remark 5.** Note that the hypotheses of the theorem are minimal. If a potential
assigner $\Psi$ is not sound, there is a game $G$ for which the algorithm returns
$\Phi \geq \Psi(G) > \text{En}_G$. If it does not satisfy (ii), there is a game for which the
algorithm stops in the first iteration, returning the potential $\Phi = 0 \neq \text{En}_G$.

**Example 6 (Simple value iteration of Brim et al. [6]).** Define the potential as-

signer $\Psi_{\text{First}^+}$ by assigning the potential $\text{First}^+_G(v)$, the first-if-positive value, to a vertex $v$. This potential is easily computed in linear time as it coincides for each
Max (resp. Min) vertex $v$, with the maximal (resp. minimal) value of $\max(w, 0)$
where $w$ ranges over outgoing weights. Clearly $\Psi_{\text{First}^+} \leq \text{En}_G$, since for any

\[4\text{By a small abuse of notation, we allow to sum potential with different domains. If } \varphi : V \rightarrow \mathbb{N}^\infty \text{ and } \varphi' : V' \rightarrow \mathbb{N}^\infty \text{ with } V' \subseteq V, \text{ then } \varphi + \varphi'(v) = \varphi(v) \text{ for all } v \notin V'.\]
sequence of weights \( w_0 w_1 \ldots \), it holds that \( \text{First}^\dagger(w_0 w_1 \ldots) \leq \text{En}(w_0 w_1 \ldots) \). Finally, if \( \Psi_{\text{First}^\dagger} = 0 \), then from any vertex Min can ensure that no positive weight is ever seen, which entails \( \text{En}_G = 0 \). We conclude that \( \Psi_{\text{First}^\dagger} \) is sound and complete; the fast value iteration algorithm coincides with that of [6].

**Example 7.** Any (determined) valuation \( \text{val}: \mathbb{Z}^\omega \rightarrow \mathbb{R}^\infty \) induces a potential assigner \( \Psi_{\text{val}} \), namely, the one that assigns to each game \( G \) the potential given by \( \text{val}_G(v) \). If the valuation satisfies \( \text{val} \leq \text{En} \) over weight sequences, then \( \Psi_{\text{val}} \) is sound. Moreover, if \( \text{val}(w_0 w_1 \ldots) > 0 \) whenever \( w_0 > 0 \), then \( \Psi_{\text{val}} \) is complete. This includes the previous example, and more interestingly, this includes the valuation \( \text{En}^\dagger \), which is the subject of Section 4.1.

Of course, an important requirement over \( \Psi \) to make Algorithm 1 relevant is that we should be able to compute \( \Psi(G) \) efficiently. Note that the potential assigner corresponding to the \( \text{En} \)-values of a game satisfies all the required hypothesis, and makes Algorithm 1 terminate in a single iteration.

**∞-attraction.** In many occurrences, the algorithm can be simplified by removing lines 6-8 and stopping when a fixpoint is reached (which can be implemented by replacing line 9 with “while \( \Psi(G) \neq \Psi(G) \)”). We say that potential assigners with this property are \( \text{∞-attracting} \). We provide easy-to-check sufficient conditions for \( \text{∞-attracting} \) in Appendix A.2.

**Modularity.** The framework of fast value iteration is specially well suited for a modular approach, allowing to solve games piecewise, as we show next. A subgame is a pair \( (G', G) \), with \( G' \subseteq G \). Let \( \mathcal{S} \) be a class of subgames. We say that a potential assigner \( \Psi \) is \( \mathcal{S} \)-sound if for all subgames \( (G', G) \in \mathcal{S} \) it holds that \( \Psi(G') \leq \text{En}_G|_{G'} \), that is, the potential is sound when applied only to this part of the game. (We note that for instance, any sound potential is \( \mathcal{S}_{\text{Trap}} \)-sound for the class of subgames \( (G', G) \) such that \( G' \) is a Min-trap.) Therefore, if \( \Psi \) is \( \mathcal{S} \)-sound, we can solve subgames in \( \mathcal{S} \) partially, and apply the corresponding potential reduction in the whole game, progressing towards a computation of the \( \text{En} \)-values.

The rest of the section is devoted to a proof of Theorem 4. Detailed proofs are available in Appendix A.1.

**Termination.** Termination of Algorithm 1 is ensured thanks to lines 6 and 7: the function \( \Phi \) strictly increases in each non-terminating iteration, and it only takes values in \( [0, nW] \cup \{\infty\} \), hence the bound \( n^2W \).

**Correctness.** We now state the key technical theorem enabling our framework. It describes the effect of sound potential reductions over energy values, allowing to combine them. From it, we easily derive compositionality of sound potentials.

**Theorem 8 (Update of energy values).** If \( \varphi \) is sound then \( \text{En}_G = \varphi + \text{En}_G \).

5Formally, reducing from complexity \( O(n^2mw) \) to \( O(nmw) \) requires some additional bookkeeping.
Corollary 9 (Compositionality). If $\varphi$ is sound for $G$ and $\varphi'$ is sound for $G_x$, then $\varphi + \varphi'$ is sound for $G$.

Proof. As $\varphi'$ is sound for $G_x$, we have that $\varphi' \leq E_{G_x}$. Adding $\varphi$ on both sides, we get $\varphi + \varphi' \leq \varphi + E_{G_x}$. By Theorem 8, the right hand-side is equal to $E_G$, as desired. \hfill \Box

We are now ready to prove Theorem 4. (The formal proof requires a bit more work regarding vertices sent to $\infty$, see Appendix A.1.)

Proof (Informal proof for Theorem 4). Let $G_i$, $\varphi_i = \Psi(G_i)$ and $\Phi_i = \varphi_1 + \cdots + \varphi_{i-1}$ denote the game, potential and cumulative sum at the $i$-th iteration of the algorithm. Since $\Psi$ is sound, $\varphi_i$ is sound for $G_i$ for all $i$. Thus it follows from an easy induction and compositionality that for all $i$, $\Phi_i$ is sound for $G$. In particular, for the maximal $i$, Theorem 8 gives $\Phi_i + E_{G_i} = E_G$, but moreover since $\varphi_i = 0$ we get by completeness that $E_{G_i} = 0$ which concludes. \hfill \Box

3.3 Asymmetry and alternating fast value iteration

Fast value iteration is based on successive underapproximations of the energy valuation $E$, which is inherently asymmetric. However, the initial problem (solving mean-payoff games) is itself symmetric, which calls for the design of more symmetrical solutions, a recurring theme in the literature [21,22,34,36].

Dual algorithm computing Max-values. Let $\overline{G}$ be the game obtained by swapping $V_{\text{Min}}$ and $V_{\text{Max}}$ and relabelling the weights by $\overline{w} = -w$. The two games are essentially equivalent, for instance the mean-payoff values in $\overline{G}$ and $G$ are opposite. However asymmetric algorithms such as value iterations behave differently over each game; this is useful for instance if one wants to compute Max strategies in $G$, which are output by running value iterations in the dual. But this still does not provide a symmetric solution.

Alternating fast value iteration. We now consider alternating versions of the algorithm, by working with potentials in $\varphi : V \to \mathbb{Z} \cup \{\pm \infty\}$. The algorithm applies potential reductions corresponding to $\Psi$ and its dualized version $\overline{\Psi}$ on the same game in an alternating fashion, until all vertices are sent to $+\infty$ or $-\infty$. Naturally, when a vertex is set to $+\infty$ or $-\infty$, the adequate attractor is computed and removed from the game.

 Assuming the potential assigner $\Psi$ is sound, since sound potential reductions do not alter winning regions, the algorithm is correct and Min’s winning region is the preimage of $-\infty$ by the final potential. Termination, however, is not easily guaranteed. Interestingly, we observe experimentally that, for some potential assigners, this alternating algorithm always terminates, and it is even remarkably fast (see Section 5). We leave as an interesting open problem to determine for which potential assigners (if any) this algorithm terminates (see conclusion).
4 Instances of fast value iteration and theoretical comparisons

We have already shown (Example 6) how SVI instantiates in our framework. In this section, we introduce further potential assigners to capture known efficient algorithms for energy games, and prove their soundness. This provides a streamlined and unified presentation of (versions of) the algorithms OSI [33] and QDPM [2] (Section 4.1), namely the positive path iteration algorithm (PPI). We also propose a dynamic variant DPPI, corresponding to a potential assigner generating potentials with provably larger values. At the end of the section we also discuss the GKK algorithm [19], and then provide formal comparisons between the four algorithms stated in our framework.

In all cases, we find that the algorithms are easier to explain over simple games, which we will assume without loss of generality (see Lemma 3); note also that simplicity is preserved by potential reductions.

4.1 The positive path iteration algorithm

We now study the fast value iteration algorithm corresponding to the potential assigner \( \Psi_{\text{En}^+} (G) = \text{En}_{\text{G}}^+ \). We call it the Positive path iteration algorithm (PPI). It is immediate to check that the potential assigner \( \Psi_{\text{En}^+} \) is sound and complete (see Example 7), so Theorem 4 applies, directly giving correctness of PPI. Moreover, we can in this case simplify the algorithm by removing lines 6-7 in Algorithm 1, because \( \Psi_{\text{En}^+} \) is \( \infty \)-attracting. We refer to Appendix A.2 for a proof of this fact.

**Proposition 10.** The potential assigner \( \Psi_{\text{En}^+} \) is \( \infty \)-attracting.

We let \( N_G \) denote the set of vertices from which Min can ensure to immediately see a negative vertex: \( v \in V_{\text{Max}} \) (resp. \( V_{\text{Min}} \)) belongs to \( N \) if and only if all outgoing edges (resp. some outgoing edge) have weight \( < 0 \). Note that computing \( \text{En}^+ \)-values in \( G \) corresponds to solving a variant of the energy game which stops whenever \( N \) is reached. It turns out that this problem is (efficiently) tractable, thanks to two-player game extensions of Dijkstra’s algorithm. In fact, two seemingly distinct algorithms are known, corresponding to OSI [33] and QPDM [2]. Remarkably, Khachiyan, Gurvich and Zhao [24] solved the same problem earlier and in a different context (with an algorithm similar to Schewe’s).

Two algorithms for computing \( \text{En}^+ \). We now describe the two algorithms, respectively extracted from [33] and [2]. We first introduce some notation. For a subset \( F \subseteq V \), a vertex \( v \in F \) and an edge \( vv' \), we define \( \text{esc}_F(vv') = w(vv') \) if \( v' \notin F \), and \( \text{esc}_F(vv') = \infty \), if \( v' \in F \). We define the escape value of a vertex as:

\[
\text{esc}_F(v) = \begin{cases} 
\min \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Min}}, \\
\max \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Max}}.
\end{cases}
\]

\[\text{esc}_F(v) = \begin{cases} 
\min \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Min}}, \\
\max \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Max}}.
\end{cases}\]

\[\text{esc}_F(v) = \begin{cases} 
\min \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Min}}, \\
\max \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Max}}.
\end{cases}\]

\[\text{esc}_F(v) = \begin{cases} 
\min \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Min}}, \\
\max \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Max}}.
\end{cases}\]

\[\text{esc}_F(v) = \begin{cases} 
\min \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Min}}, \\
\max \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Max}}.
\end{cases}\]

\[\text{esc}_F(v) = \begin{cases} 
\min \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Min}}, \\
\max \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Max}}.
\end{cases}\]

\[\text{esc}_F(v) = \begin{cases} 
\min \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Min}}, \\
\max \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Max}}.
\end{cases}\]

\[\text{esc}_F(v) = \begin{cases} 
\min \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Min}}, \\
\max \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Max}}.
\end{cases}\]

\[\text{esc}_F(v) = \begin{cases} 
\min \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Min}}, \\
\max \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Max}}.
\end{cases}\]

\[\text{esc}_F(v) = \begin{cases} 
\min \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Min}}, \\
\max \{ \text{esc}_F(vv') \mid w(vv') \geq 0 \}, & \text{if } v \in V_{\text{Max}}.
\end{cases}\]
We will only consider subsets $F \subseteq N_G$, so Max vertices have a non-negative outgoing edge and Min vertices have only non-negative outgoing edges, in particular $esc_F(v) \geq 0$. It can be seen as the minimal weight that Min can force to see while leaving $F$ immediately from $v$, or $\infty$ is she cannot force to leave $F$ in one step, assuming Max is constrained to playing non-negative edges. We further let $F^{<\infty}$ denote the set of vertices with finite $esc_F$, and $F^{<\infty}_\text{Max}$ and $F^{<\infty}_\text{Min}$ their intersections with $V_{\text{Max}}$ and $V_{\text{Min}}$. Last, for $v \in V$, the notation $\varphi_v(v) \leftarrow x$ indicates that $\varphi_v$ is the potential defined by $\varphi_v(v) = x$ and $\varphi_v(v') = 0$ for $v' \neq v$.

### Algorithm 2 Subprocedure in OSI

**Input:** Simple game $G$

$\Phi \leftarrow 0$

$F \leftarrow V \setminus N_G$

while $F^{<\infty} \neq \emptyset$

if $F^{<\infty}_\text{Max} \neq \emptyset$

let $v \in F^{<\infty}_\text{Max}$

$\Phi(v) \leftarrow \max_{w(vv') \geq 0} w(vv') + \Phi(v')$

else if $F^{<\infty}_\text{Min} \neq \emptyset$

let $v \in F^{<\infty}_\text{Min}$ minimizing

$m = \min_{v' \in F^{<\infty}_\text{Min}} w(vv') + \Phi(v')$

$\Phi(v) \leftarrow m$

end if

end while

$\varphi_v(v) \leftarrow \infty$ for all $v \in F$

return $\Phi$

### Algorithm 3 Subprocedure in QDPM

**Input:** Simple game $G$

$\Phi = 0$

$F \leftarrow V \setminus N_G$

while $F^{<\infty} \neq \emptyset$

let $v \in \arg\min_{v} esc_F(v)$

$\varphi_v(v) \leftarrow esc_F(v)$

$\Phi \leftarrow F \setminus \{v\}$

$G \leftarrow G_{\varphi_v}$

$\Phi = \Phi + \varphi_v$

end while

$\Phi_e(v) \leftarrow \infty$ for all $v \in F$

return $\Phi$

**Theorem 11 (Adapted from [24,33,2]).** Algorithms 2 and 3 both compute $En^+_G$, and both can be implemented to run in $O(m + n \log n)$ operations.

A detailed proof is given in Appendix A.3. In Appendix B we include detailed comparisons between the Positive path iteration algorithm (PPI), and the algorithms OSI and QDPM.

### 4.2 A new fast value iteration algorithm

Drawing inspiration from Algorithms 2 and 3 above, we introduce another potential assigner, leading to a fast value iteration algorithm which we call *Dynamic positive path iteration (DPPI)*. Note that both algorithms above compute the Min attractor to $N_G$ over non-negative edges, which corresponds exactly to the set of vertices with finite $En^+$, and obtain the values of $En^+$ by backtracking. We will also backtrack over the same attractor, and just as in Algorithm 3, we make potential updates on the fly. The difference is in the precise way in which we choose the vertices, which enables in our case that some of the potential updates may cause new edges to become positive, which will then be taken into account, sometimes leading to a potential $> En^+$. 


Algorithm 4 Computation of the $\Psi_{\text{DPPI}}$-potential.

Input: Simple game $G$

1: $F \leftarrow V \setminus N_G$
2: while $F < \infty \neq \emptyset$ do
3: if there is $v \in \arg\min_{v \in V_{\text{Min}}} \text{esc}_F(v) \cap V_{\text{Min}}$ then
4: $\varphi_v(v) \leftarrow \text{esc}_F(v)$
5: else let $v \in \arg\max_{v \in F < \infty} \text{esc}_F(v)$ and $\varphi_v(v) \leftarrow \text{esc}_F(v)$
6: end if
7: $F \leftarrow F \setminus \{v\}$
8: $G \leftarrow G_{\varphi_v}$
9: $\varphi_v(v) \leftarrow \infty$ for all $v \in F$
10: return $\sum_{v \in V} \varphi_v$

Lemma 12. The potential assigner $\Psi_{\text{DPPI}}$ is sound and complete for simple games.

We see DPPI as a marginal improvement over PPI, but an improvement nonetheless, showing that the barrier imposed by PPI can be broken, motivating future work. Figure 3 shows a game where DPPI performs fewer iterations than PPI, while Lemma 13 below proves that for any game $G$, $\Psi_{\text{DPPI}}(G) \geq \Psi_{\text{En}^+}(G)$.

![Fig. 3. A game $G$ (all vertices belong to Max) where DPPI performs a single iteration, as $\Psi_{\text{DPPI}}(G) = [v_1 \to 10; v_2 \to 9; v_N \to 0] = \text{En}_G$. In contrast, PPI requires two iterations since $\text{En}^+ = [v_1 \to 10; v_2 \to 5; v_N \to 0]$.](image)

4.3 The GKK algorithm

We include a short discussion about the GKK algorithm; a more detailed modern exposition, including state-of-the-art upper bounds and comparison with the related approach of Dorman et al. [11], was proposed by Ohlmann [31].

The GKK algorithm is the $\Psi_{\text{GKK}}$-fast value iteration where $\Psi_{\text{GKK}}$ is the potential assigner defined as follows.

Let $V_-$ be the set of vertices from which Min can ensure that a negative edge is seen before the first positive edge. (Note that $V_-$ coincides with $(\text{En}_G^+)^{-1}(0)$.)

Likewise, let $V_+$ denote the set of vertices from which Max can ensure seeing a positive edge before a negative one; and observe that in a simple game, $V_+$ is the complement of $V_-$. Consider the maximal value $w_+$ such that from any
Fast value iterations for energy games

vertex of $V_+$. Max can ensure to add up to $w_+$ before a negative weight is seen (alternatively, $w_+$ is the smallest nonzero value of $E_{G_+}$); and dually for $w_-$. Note that, if from any vertex in $V_+$, Max can ensure to remain in $V_+$ while seeing positive vertices, then $w_+ = \infty$. Clearly $w_+ \leq E_{G_+} \leq E_{G}$ over $V_+$. We define $\Psi_{GKK}(G)(v)$ to be $\min(w_+, -w_-)$ if $v \in V_+$ and $0$ otherwise. Soundness follows from the inequality above, and completeness is easy to prove. Moreover, $\Psi_{GKK}$ is $\infty$-attracting (a proof of this fact is included in Appendix A.2).

The potential $\Psi_{GKK}$ has a remarkable symmetric property: the assigned potentials are the same over $G$ and over its dual $\overline{G}$: $\Psi_{GKK} = \overline{\Psi_{GKK}}$.\footnote{This was first observed by Ohlmann [31] leading to an improved upper bound.} In particular, the algorithm and its alternating version coincide.

### 4.4 Comparing fast value iteration algorithms

We now propose formal comparisons between the above potential assigners. Intuitively, in order to minimise the number of iterations of a fast value iteration algorithm, we should seek for potentials assigning large values to vertices, so that a “big step” is produced in each iteration. In this sense, if $\Psi \leq \Psi'$, the $\Psi'$-FVI algorithm is expected to perform better. A priori, the sequence of games produced by the two algorithms will diverge, impeding formal comparisons on the number of iterations. However, for monotone potential assigners, we can also compare the number of iterations of the induced FVI algorithms.

**Lemma 13.** For every game $G$,

$$\Psi_{First}^+(G) \leq \Psi_{En}^+(G) \leq \Psi_{DPPI}(G), \quad and \quad \Psi_{GKK}(G) \leq \Psi_{En}(G).$$

Moreover, there are games making these inequalities strict. The potential assigners $\Psi_{First}^+$ and $\Psi_{GKK}$ are incomparable.

Let $\Psi, \Psi'$ be two potential assigners. We say that $\Psi'$ is monotonically larger than $\Psi$, noted $\Psi \preceq_{\text{mon}} \Psi'$ if, for all game and potentials $\varphi \leq \varphi'$, it holds

$$\varphi + \Psi(G_{\varphi}) \leq \varphi' + \Psi'(G_{\varphi'}).$$

We say that $\Psi$ is monotone if $\Psi \preceq_{\text{mon}} \Psi$. The next two lemmas are immediate.

**Lemma 14.** Let $\Psi, \Psi'$ be sound, complete potential assigners, and assume $\Psi \preceq_{\text{mon}} \Psi'$. Then over any input game $G$, the $\Psi'$-FVI algorithm terminates in less iterations than the $\Psi$-FVI algorithm.

**Lemma 15.** Let $\Psi, \Psi_1, \Psi_2$ be potential assigners. It holds:

$$\Psi \preceq_{\text{mon}} \Psi_1 \leq \Psi_2 \implies \Psi \preceq_{\text{mon}} \Psi_2.$$ 

In particular, if $\Psi$ is monotone and $\Psi \preceq \Psi'$, then $\Psi \preceq_{\text{mon}} \Psi'$.\footnote{This was first observed by Ohlmann [31] leading to an improved upper bound.}
Proposition 16. The potential assigner $\Psi_{\text{First}^+}$ is monotone. Therefore, PPI and DPPI terminate in less iterations than SVI over any input game.

Proof. Let $\varphi \leq \varphi'$ be two potentials on $G$. Let $v$ be a Min-vertex (the proof for Max-vertices is the same). It suffices to remark that:

$$\varphi(v) + \Psi_{\text{First}^+}(G\varphi)(v) = \min_{v' \in E} \varphi(v') + w(vv') \leq \max_{v' \in E} \varphi'(v') + w(vv').$$

$\square$

An interesting open question is whether the potential $\Psi_{\text{GKK}}$ is monotone.

5 Experimental results

We focus on two distinct game-solving applications: energy game solving, which is the natural target for our algorithms, and parity game solving, which incurs a conversion cost to energy games but allows using established parity game benchmarks and comparison with other parity game solvers.

After explaining the technical aspects of our implementation, and choices of algorithms and benchmarks, we discuss the most remarkable behaviours that can be observed in the experiments.

The algorithms were implemented in Oink [9], a tool providing a uniform framework for the comparison of parity game solvers. Our implementation can be obtained at: https://github.com/michaelcadilhac/oink/tree/TACAS25.

Experiments were carried on an Intel® Core™ i7-8700 CPU @ 3.20GHz paired with 16GiB of memory, each test being capped at 60 seconds and 10GiB of memory. Arithmetic operations over multiple precision integers are carried out using the GNU Multiple Precision Arithmetic library (GMP). All games are available at: https://github.com/michaelcadilhac/game-benchmarks/tree/TACAS25.

Set of algorithms. We compare our implementations of PPI, DPPI and their alternating versions (PPI-alt and DPPI-alt)\(^8\) to 4 other algorithms: QDPM from [2], Zielonka’s recursive algorithm (ZLK), Tangle learning (TL) and Recursive tangle learning (RTL) (winner of the latest edition of SYNTCOMP) from [8,9]. Only one of them (QDPM) can be executed over general energy games, the other three are parity game solvers.

We remark that we do not include comparisons with SVI [6], nor with GKK-DKZ [19,11], as these algorithms are known to be inefficient in practice [2] and incur frequent timeouts. Also, we have not compared to an independent implementation of OSI, as we have not found one such implementation computing winning regions consistent with the rest of the algorithms.

\(^8\)In favour of clarity, we omit the DPPI-alt plots, as they perform identically to PPI-alt. This is expected, given the similarity of the plots of PPI and DPPI.
5.1 Parity game solving

We show the results of our experiments on parity games in Figure 4. We rely on the yearly competition SYNTCOMP24 for our benchmarks, which has a competition track for parity game solvers, and on the benchmark suite of Keiren [23]. We subdivide the 779 benchmarks into two categories: synthetic games (crafted by researchers, usually with the intent of being hard for certain solving approaches) and organic games (the natural counterpart of the synthetic games). We note that the synthetic games include the “two counter games” examples [10], in which TL and RTL show an exponential behaviour. It also contains the family of examples by Friedmann [17], exponential for OSI. (We refer to Appendix C for more details on Friedmann’s family of examples.) The organic games are essentially the ones provided by Keiren [23], see therein for their origin.

As is usual in this settings we present the experimental results as a survival plot, which indicates how many tests are solved (x-axis) within a time limit (y-axis, time per test). In order to solve input parity games with energy games solvers, we first need to convert the parity game into an energy one. This step is rather costly, as the priorities of the parity game suffer an exponential blow-up when converted to weights of an energy game. This cost is included in the runtime of our algorithms as well as QDPM.

5.2 Energy game solving

We show the results of our experiments on parity games in Figure 4. We modified Oink so that it would accept negative weights and implemented a strategy-checker for energy games — this boils down to checking that, in the game restricted to the strategy, Max-winning strongly-connected components do not have infinite negative cycles, and symmetrically for Min.
We consider randomly generated bipartite graphs. The restriction to bipartite graphs is justified by the fact that, otherwise, the vast majority of vertices are part of winning cycles controlled by the same player, making the game (and its resolution) much easier. We separate instances that are sparse (the out-degree of each vertex is 2) or dense (the number of edges is $n^2/5$).

![Survival plot for energy games benchmarks, divided in sparse and dense.](image)

**Fig. 5.** Survival plot for energy games benchmarks, divided in sparse and dense.

### 5.3 Conclusions of the experiments

In light of the experiments above, we derive the following conclusions.

1. Overall, the fast value iteration framework captures several algorithms (PPI, PPI-alt, QDPM) that perform competitively in standard benchmarks of parity games. Despite being less efficient than leading parity game solvers, they are remarkably robust against hard instances, particularly PPI-alt.

2. The alternating version of PPI and DPPI, for which we were unable to prove termination in theory, always terminate. Moreover, over instances coming from parity games benchmarks, they are significantly faster than their asymmetric counterparts.

3. While DPPI was introduced as a theoretically enhanced version of PPI, there is no significant difference in the running time of these algorithms. In fact, DPPI tends to be slightly slower, due to the increased cost in the computation of the potential.

4. Although based on the same algorithmic ideas, QDPM consistently outperforms PPI, by almost an order of magnitude. This difference can be explained by two factors: (1) QDPM uses some smart implementation optimizations [2, Sect. 5], and (2) our implementation of PPI is tailored for (usual) edge-weighted games, whereas QDPM is implemented for vertex-labelled game
(for which two weights outgoing a given vertex are always equal). Details and estimates on why and how this difference may affect the performance of the algorithms is discussed in detail in Appendix B.4.

6 Conclusion and future work

We have presented a general framework to describe algorithms for energy games, capturing and providing simple descriptions and correctness proofs for many of them, including the top performing ones in practice. The fast value iteration framework raises numerous exciting questions; we outline some of them here.

New algorithms. The new framework provides a very easy way to propose new correct algorithms: it suffices to define a potential assigner which is sound, complete, and computable in polynomial time. We have isolated $\Psi_{En^+}$ as an important potential assigner, implicitly used by the two fastest algorithms solving energy games, and presented the potential $\Psi_{DPPI}$ which, while still being computable in polynomial time, is $\geq \Psi_{En^+}$ in general.

Question 17. Does there exist a reasonable\footnote{A non-reasonable example meeting the requirements is $\Psi_{DPPI} \circ \Psi_{DPPI}$.} potential assigner which is sound, complete, computable in polynomial time and $\geq \Psi_{DPPI}$?

Alternating algorithms. Our framework also allows to design symmetric alternating algorithms, for which we are unable to prove termination using the currently available tools. Our empirical study shows that, in practice, these not only terminate, but are often considerably faster than their asymmetric counterparts.

Question 18. Do alternating fast value iterations terminate over simple games?

We stress the fact that the question is open for all sound and complete potential assigners (except for GKK, for which the alternating algorithm coincides with the normal one, see Section 4.3).

Lower bounds. Friedmann proposed notoriously involved constructions which provide exponentially many iterations for strategy improvement algorithms in [17]. (We discuss in detail Friedmann’s family of examples in Appendix C.) Although these include OSI (see [17, Sect. 4.6.2]), our experiments show that PPI can solve these instances in linear time, and PPI-alt in a constant number of 2 iterations. Currently, we lack any family of examples in which PPI takes more than a linear number of iterations, although we expect that it should admit exponential lower bounds.

Question 19. Can one design superpolynomial lower bounds on the number of iterations for PPI? And (more challenging) for its alternating variant?
Randomized initialization. As remarked in Section 3.1, the weights of the cycles of $G$ and $G_\varphi$ coincide for any finite potential $\varphi$, so the threshold problem for the mean-payoff objective is equivalent over these games. Therefore, we can initialize a given game with an arbitrarily potential $\varphi$, and solve the “perturbed game”. This directly provides a randomized version of any algorithm: add a random perturbation before execution. This idea is not novel, it was studied empirically by Beffara and Vorobyov [1] for the GKK algorithm; and lower bounds were later derived by Lebedev [25] for the same algorithm.

Question 20. Is the randomized variant of PPI subexponential? More generally, can we design a potential assigner whose associated randomized fast value iteration is subexponential?

Smooth analysis. An interesting parallel can be drawn with smooth analysis [35], which consider small perturbations of the input (randomized initialization is difference since we get an equivalent input). In fact, it was recently established that there is a strategy improvement algorithm for mean-payoff games that is polynomial in the sense of smooth analysis [26].

Question 21. Can the algorithm of [26] be recast as a fast value iteration?

Acknowledgments. We thank the authors of [2] for kindly providing their implementation of QDPM and Alexander Kozachinskyi for pointing out to us several important references.

Antonio Casares is supported by the Polish National Science Centre (NCN) grant “Polynomial finite state computation” (2022/46/A/ST6/00072).

References

1. Beffara, E., Vorobyov, S.: Is randomized Gurvich-Karzanov-Khachiyan’s algorithm for parity games polynomial? In: Technical report 2001-025. Uppsala University, Sweden (2001)
2. Benerecetti, M., Dell’Erba, D., Mogavero, F.: Solving mean-payoff games via quasi dominions. Inf. Comput. 297, 105151 (2024). https://doi.org/10.1016/J.IC.2024.105151
3. Björklund, H., Sandberg, S., Vorobyov, S.G.: A combinatorial strongly subexponential strategy improvement algorithm for mean payoff games. In: MFCS. Lecture Notes in Computer Science, vol. 3153, pp. 673–685. Springer (2004). https://doi.org/10.1007/978-3-540-28629-5_52
4. Björklund, H., Vorobyov, S.G.: Combinatorial structure and randomized subexponential algorithms for infinite games. Theor. Comput. Sci. 349(3), 347–360 (2005)
5. Bouyer, P., Fahrenberg, U., Larsen, K.G., Markey, N., Srba, J.: Infinite runs in weighted timed automata with energy constraints. In: FORMATS. Lecture Notes in Computer Science, vol. 5215, pp. 33–47. Springer (2008)
6. Brim, L., Chaloupka, J., Doyen, L., Gentilini, R., Raskin, J.: Faster algorithms for mean-payoff games. Formal Methods in System Design 38(2), 97–118 (2011)
7. Calude, C.S., Jain, S., Khoussainov, B., Li, W., Stephan, F.: Deciding parity games in quasipolynomial time. In: STOC. pp. 252–263 (2017)
8. van Dijk, T.: Attracting tangles to solve parity games. In: CAV. Lecture Notes in Computer Science, vol. 10982, pp. 198–215. Springer (2018). https://doi.org/10.1007/978-3-319-96142-2_14
9. van Dijk, T.: Oink: An implementation and evaluation of modern parity game solvers. In: TACAS. Lecture Notes in Computer Science, vol. 10805, pp. 291–308. Springer (2018). https://doi.org/10.1007/978-3-319-89960-2_16
10. van Dijk, T.: A parity game tale of two counters. In: GandALF. EPTCS, vol. 305, pp. 107–122 (2019). https://doi.org/10.4204/EPTCS.305.8
11. Dorfman, D., Kaplan, H., Zwick, U.: A faster deterministic exponential time algorithm for energy games and mean payoff games. In: ICALP. pp. 114:1–114:14 (2019)
12. Ehrenfeucht, A., Mycielski, J.: Positional strategies for mean payoff games. International Journal of Game Theory 109(8), 109–113 (1979)
13. Emerson, E.A., Jutla, C.S.: Tree automata, $\mu$-calculus and determinacy. In: FOCS. pp. 368–377. IEEE Computer Society (1991)
14. Fearnley, J.: Non-oblivious strategy improvement. In: LPAR. Lecture Notes in Computer Science, vol. 6355, pp. 212–230. Springer (2010). https://doi.org/10.1007/978-3-642-17511-4_13
15. Fijalkow, N., Gawrychowski, P., Ohlmann, P.: Value iteration using universal graphs and the complexity of mean payoff games. In: MFCS. LIPIcs, vol. 170, pp. 34:1–34:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2020)
16. Fredman, M.L., Tarjan, R.E.: Fibonacci heaps and their uses in improved network optimization algorithms. In: FOCS. pp. 338–346. IEEE Computer Society (1984)
17. Friedmann, O.: Exponential Lower Bounds for Solving Infinitary Payoff Games and Linear Programs. Ph.D. thesis, Ludwig Maximilians University Munich (2011), http://edoc.ub.uni-muenchen.de/13294/
18. Gallai, T.: Maximum-minimum sätze über graphen. Acta Math. Acad. Sci. Hung. (9), 395–434 (1958)
19. Gurvich, V.A., Karzanov, A.V., Khachiyan, L.G.: Cyclic games and an algorithm to find minimax cycle means in directed graphs. USSR Computational Mathematics and Mathematical Physics 28, 85–91 (1988)
20. Jacobs, S., Perez, G.A., Abraham, R., Bruyere, V., Cadilhac, M., Colange, M., Delfosse, C., van Dijk, T., Duret-Lutz, A., Faymonville, P., Finkbeiner, B., Khalimov, A., Klein, F., Luttenberger, M., Meyer, K., Michaud, T., Pommellet, A., Renkin, F., Schlehuber-Caissier, P., Sakr, M., Sickert, S., Stacquet, G., Tamines, C., Tentrup, L., Walker, A.: The reactive synthesis competition (SYNTCOMP): 2018-2021 (2022)
21. Jurdziński, M., Morvan, R., Ohlmann, P., Thejaswini, K.S.: A symmetric attractor-decomposition lifting algorithm for parity games. CoRR abs/2010.08288 (2020), https://arxiv.org/abs/2010.08288
22. Jurdzinski, M., Morvan, R., Thejaswini, K.S.: Universal algorithms for parity games and nested fixpoints. In: Principles of Systems Design - Essays Dedicated to Thomas A. Henzinger on the Occasion of His 60th Birthday. Lecture Notes in Computer Science, vol. 13660, pp. 252–271. Springer (2022). https://doi.org/10.1007/978-3-031-22337-2_12
23. Keiren, J.J.A.: Benchmarks for parity games. In: Dastani, M., Sirjani, M. (eds.) Fundamentals of Software Engineering, pp. 127–142. Springer International Publishing, Cham (2015)
24. Khachiyan, L., Gurvich, V., Zhao, J.: Extending Dijkstra’s algorithm to maximize
the shortest path by node-wise limited arc interdiction. In: CSR. Lecture Notes in
Computer Science, vol. 3967, pp. 221–234. Springer (2006)
25. Lebedev, V.: Exponential examples of solving parity games. Computational Math-
ematics and Mathematical Physics 56, 688–697 (2016)
26. Loff, B., Skomra, M.: Smoothened analysis of deterministic discounted
and mean-payoff games. In: ICALP. LIPIcs, vol. 297, pp. 147:1–147:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2024).
https://doi.org/10.4230/LIPICS.ICALP.2024.147
27. Luttenberger, M.: Strategy iteration using non-deterministic strategies for solving
parity games. CoRR (2008), http://arxiv.org/abs/0806.2923
28. Luttenberger, M., Meyer, P.J., Sickert, S.: Practical synthesis of reactive systems
from LTL specifications via parity games. Acta Informatica 57(1-2), 3–36 (2020)
29. Meyer, P.J., Sickert, S., Luttenberger, M.: Strix: Explicit reactive synthesis strikes
back! In: CAV. Lecture Notes in Computer Science, vol. 10981, pp. 578–586.
Springer (2018)
30. Mostowski, A.W.: Games with forbidden positions. Tech. Rep. 78, University of
Gdansk (1991)
31. Ohlmann, P.: The GKK algorithm is the fastest over simple mean-payoff games.
In: CSR. Lecture Notes in Computer Science, vol. 13296, pp. 269–288. Springer
(2022)
32. Puri, A.: Theory of Hybrid Systems and Discrete Event Systems. Ph.D. thesis,
EECS Department, University of California, Berkeley (dec 1995)
33. Schewe, S.: An optimal strategy improvement algorithm for solving parity and
payoff games. In: CSL. Lecture Notes in Computer Science, vol. 5213, pp. 369–
384. Springer (2008)
34. Schewe, S., Trivedi, A., Varghese, T.: Symmetric strategy improvement. In: ICALP.
Lecture Notes in Computer Science, vol. 9135, pp. 388–400. Springer (2015).
https://doi.org/10.1007/978-3-662-47666-6_31
35. Spielman, D.A., Teng, S.: Smoothened analysis of algorithms: Why the simple-
plex algorithm usually takes polynomial time. J. ACM 51(3), 385–463 (2004).
https://doi.org/10.1145/990308.990310
36. Thejaswini, K.S., Ohlmann, P., Jurdzinski, M.: A technique to speed up symmet-
ric attractor-based algorithms for parity games. In: FSTTCS. LIPIcs, vol. 250,
pp. 44:1–44:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2022).
https://doi.org/10.4230/LIPIICS.FSTTCS.2022.44
A Correctness of algorithms

A.1 Correctness of the fast value iteration meta-algorithm

First, observe that for a finite path \( \pi = v_0 \rightarrow v_1 \rightarrow \ldots \rightarrow v_k \) which visits only vertices with finite potential, its sum in \( G_\varphi \) is given by

\[
\text{sum}_\varphi(\pi) = \text{sum}(\pi) - \varphi(v_0) + \varphi(v_k).
\]

We start with a technical lemma.

**Lemma 22.** Let \( \sigma_0 \) be an \( \text{En} \)-optimal Min strategy in \( G \) and \( \pi = v_0 \rightarrow v_1 \rightarrow \ldots \rightarrow v_k \) be a finite path consistent with \( \sigma_0 \) such that \( \text{En}_G(v_k) < \infty \). Then we have

\[
\text{sum}(\pi) \leq \text{En}_G(v_0) - \text{En}_G(v_k).
\]

**Proof.** Let \( \pi' \) be an infinite path from \( v_k \) consistent with \( \sigma_0 \) and such that \( \text{En}_G(v_k) = \text{En}(w(\pi')) \). Then \( \pi\pi' \) is consistent with \( \sigma_0 \) thus \( \text{En}_G(v_0) \geq \text{En}(w(\pi')) \) by optimality. We then obtain

\[
\text{En}_G(v_0) \geq \text{En}(w(\pi')) = \sup_{\pi' \geq 0} (\text{sum}(\pi') + \text{sum}(\pi')) = \text{sum}(\pi) + \text{En}(w(\pi')) = \text{sum}(\pi) + \text{En}_G(v_k).
\]

\( \square \)

We are now ready to prove Theorem 8, which we first restate for convenience.

**Theorem 8 (Update of energy values).** If \( \varphi \) is sound then \( \text{En}_G = \varphi + \text{En}_{G,\varphi} \).

**Proof.** Let \( \varphi : V \rightarrow \mathbb{N}^{\infty} \) be a potential such that \( \varphi \leq \text{En}_G \); we aim to prove that \( \text{En}_G = \varphi + \text{En}_{G,\varphi} \) over \( V \). Consider first a vertex \( v \) with \( \text{En}_G(v) = \infty \), fix an optimal Max strategy \( \tau_0 \) in \( G \) and an infinite path \( \pi = e_0 e_1 \cdots = v_0 \rightarrow v_1 \rightarrow \ldots \) consistent with \( \tau_0 \) from \( v \); by definition we have \( \text{En}(w(\pi)) = \sup_k \sum_{i=0}^{k-1} w(e_i) = \infty \). We claim that \( \text{En}(w(\varphi(\pi))) = \infty \) which proves the wanted equality over \( v \) (both terms are infinite).

- If for some \( i \), \( w(e_i) = \infty \) then \( w(\varphi(e_i)) = \infty \) which implies the result.
- If for some \( i \), \( \varphi(v_i) = \infty \) then again we have \( w(\varphi(e_i)) = \infty \).
- Otherwise, we have for all \( k \)

\[
\text{sum}_\varphi(\pi_{<k}) = \varphi(v_k) - \varphi(v_0) + \text{sum}(\pi_{<k}),
\]

and therefore \( \sup_k \text{sum}_\varphi(\pi_{<k}) = \sup_k \text{sum}(\pi_{<k}) = \infty \), the wanted result.
We now consider a vertex \( v \) such that \( \text{En}_G(v) < \infty \). Consider an \( \text{En} \)-optimal Min strategy \( \sigma_0 : V_{\text{Min}} \to E \) in \( G \) and let \( \pi = v_0 \to v_1 \to \ldots \) be an infinite path consistent with \( \sigma_0 \) starting from \( v_0 = v \). Note that for any \( k \geq 0 \), \( v_k \) has finite energy value, and thus we obtain thanks to Lemma 22 and the hypothesis \( \varphi \leq \text{En}_G \) that

\[
\sum_{\pi} (\pi < k) = \sum (\pi < k) + \varphi(v_k) - \varphi(v_0) \\
\leq \text{En}_G(v_0) - \sum_{\pi} (\pi < k) + \varphi(v_k) - \varphi(v_0) \\
\leq \text{En}_G(v_0) - \varphi(v_0),
\]

hence \( \text{En}_\varphi(v_0) = \sup_{\pi \models \varphi} \sum_{\pi} (\pi < k) \leq \text{En}_G(v_0) - \varphi(v_0) \).

For the other inequality, consider an optimal Min strategy \( \sigma_\varphi \) in \( G_\varphi \), and let \( \pi \) be an infinite path from \( v_0 = v \) consistent with \( \sigma_\varphi \). By applying Lemma 22 in \( G_\varphi \) we now get

\[
\sum (\pi < k) = \sum_{\pi} (\pi < k) - \varphi(v_k) + \varphi(v_0) \\
\leq \text{En}_{\varphi}(v_0) - \text{En}_{\varphi}(v_k) - \varphi(v_k) + \varphi(v_0) \\
\leq \text{En}_{\varphi}(v_0) + \varphi(v_0),
\]

and the wanted result follows by taking a supremum. \( \square \)

When considering a map \( \varphi : S \to \mathbb{N}^\infty \) defined over a subset \( S \) of the vertices, we let \( \overline{\varphi} \) denote the extension of \( \varphi \) to \( V \) defined by setting its value to be \( \infty \) on \( S^c \). We require an additional technical lemma.

**Lemma 23.** Let \( G \) be a game, let \( S \) be a set of vertices with energy value \( \infty \) in \( G \), let \( A = \text{Attr}_{G}^{\text{Max}}(S) \) and let \( G' = G \setminus A \). Then \( \text{En}_G(A) = \infty \) and \( \text{En}_{G'} \leq \text{En}_G \).

In fact, the last inequality is even an equality, but we only require this direction.

**Proof.** By a strategy forcing to first go to \( S \), then following an \( \text{En} \)-optimal strategy, \( \text{Max} \) ensures energy-value \( \infty \) over \( A \). In particular, we get that \( \text{En}_{G'} \) and \( \text{En}_G \) both have value \( \infty \) over \( A \). Now since \( A \) is a \( \text{Max} \)-attractor, any Min strategy over \( A^c \) forces the game to stay in \( A^c \) therefore \( \text{En}_{G'} \leq \text{En}_G|_{A^c} \), which concludes the proof. \( \square \)

We are now ready to prove Theorem 4. The proof requires a bit of bookkeeping regarding vertices sent to \( \infty \).

**Theorem 4.** Let \( \Psi \) be a sound and complete potential assigner. Then Algorithm 1 terminates in at most \( n^2W \) iterations, and returns \( \Phi = \text{En}_G \).

**Proof.** Let \( G_0 = G \), \( A_0 = \emptyset \), \( \Phi_0 = 0 \) and for each iteration \( i = 1, 2, \ldots \) let \( G_i, \Phi_i \) and \( \psi_i \) be the values of the corresponding variables after line 4, and let \( A_i \) be computed on line 7. Let \( V_i \) be the vertex set of \( G_i \) so that \( V_i^c = A_0 \cup \cdots \cup A_{i-1} \). By definition, we have

\[
\varphi_i = \Psi(G_i), \quad \Phi_i = \Phi_{i-1} + \overline{\varphi_i}, \quad \text{and} \quad G_i = (G_{i-1})_{\varphi_{i-1}} \setminus A_{i-1}.
\]
We prove by induction on \( i \) that for all \( i \), \( \Phi_1 \) is sound for \( \mathcal{G} \) and that \( \text{En}_G(A_i) \subseteq \{ \infty \} \). For \( i = 0 \) there is nothing to prove, so we let \( i > 0 \) and assume the result known for \( j < i \).

Let \( \overline{\mathcal{G}_i} \) denote \( \mathcal{G}_{\Phi_{i-1}} \). Then \( \mathcal{G}_i = \overline{\mathcal{G}_i} \setminus \text{Attr}^{\text{Max}}(\mathcal{V}^c_i) \) and we know by induction that \( \text{En}_G \) is \( \infty \) over \( \mathcal{V}^c = A_0 \cup \cdots \cup A_{i-1} \). Thus Lemma 23 gives \( \text{En}_{G_i} \leq \text{En}_{\overline{G}_i} \). Now since \( \Psi \) is sound we have \( \mathcal{V}_i \leq \text{En}_{\overline{G}_i} \), which implies \( \overline{\mathcal{V}_i} \leq \text{En}_{\mathcal{G}_i} = \text{En}_{\overline{G}_i} \), and thus \( \overline{\mathcal{V}_i} \) is sound for \( \overline{\mathcal{G}_i} \). Thanks to the induction hypothesis and compositionality we deduce that \( \Phi_1 = \Phi_{i-1} + \varphi_i \) is sound for \( \mathcal{G} \).

There remains to prove that \( \text{En}_G(A_i) \subseteq \{ \infty \} \). Let \( v \in \Psi_i^{-1}(\{ (n-1)W + 1, \infty \}) \). Then \( \text{En}_G(v) \geq \Psi_i(v) \) therefore by Corollary 2 we get \( \text{En}_G(v) = \infty \), and conclude thanks to Lemma 7.

\[ \square \]

**A.2 A sufficient condition for \( \infty \)-attraction**

Recall that \( N_\mathcal{G} \) is the set of vertices such that Min can force to immediately see a negative edge. We say that a potential assigner is \( N \)-null if for any \( \mathcal{G} \), \( \Psi(\mathcal{G})(N_\mathcal{G}) = 0 \), that it is \( N \)-shrinking if for all \( \mathcal{G} \) it satisfies \( N_{\Psi(\mathcal{G})} \subseteq N_\mathcal{G} \), and that it is path-based if finite values of \( \Psi(\mathcal{G})(v) \) are upper bounded by the weight of a simple path from \( v \) to a vertex in \( N_\mathcal{G} \).

**Theorem 24.** Any potential assigner which is path-based and \( N \)-shrinking is \( \infty \)-attracting.

**Proof.** Consider such a potential assigner \( \Psi \) and a game \( \mathcal{G}_0 \). For all \( i \), let \( \varphi_i = \Psi(\mathcal{G}_i), \mathcal{G}_{i+1} = (\mathcal{G}_i)_{\varphi_i} \) and \( \Phi_i = \varphi_0 + \cdots + \varphi_{i-1} \).

Since \( \Psi \) is \( N \)-shrinking, we get (with obvious notations) \( N_0 \supseteq N_1 \supseteq \cdots \) and therefore since it is moreover \( N \)-null, vertices \( v' \) in \( N_i \) satisfy \( \Phi_i(v') = 0 \). Now if \( v \) is a vertex such that \( \varphi_i(v) \) is finite, then since \( \Psi \) is path based, there is a simple path \( \pi = v_0 \rightarrow \cdots \rightarrow v_k = v' \in N_i \) in \( \mathcal{G}_0 \) from \( v \) whose \( \Phi_j \)-modified sum satisfies \( \sum_{\Phi_j(\pi)} \geq \varphi_j(v) \). This rewrites as

\[
\varphi_j(v) \leq -\Phi_j(v) + \Phi_j(v') + \sum_{i=0}^{k-1} w_{\Phi_j}(v_i, v_{i+1}) \leq (n-1)W,
\]

and thus \( \Phi_j(v) \leq (n-1)W \). Stated differently, finite values remain \( \leq (n-1)W \), which guarantees termination in at most \( O(n^2W) \) iterations.

There remains to see that \( \Psi_{\text{En}^+}, \Psi_{\text{DPPI}} \) and \( \Psi_{\text{GKK}} \) satisfy the hypotheses of Theorem 24. It is obvious that they are \( N \)-null and path-based.

**Lemma 25.** The potential assigners \( \Psi_{\text{En}^+}, \Psi_{\text{DPPI}} \) and \( \Psi_{\text{GKK}} \) are \( N \)-shrinking.

**Proof.** Let \( v \notin N_\mathcal{G} \). Then observe that for all \( X \in \{ \text{En}^+, \text{DPPI}, \text{GKK} \} \), it holds that if \( v \in V_{\text{Max}} \) (resp. \( v \in V_{\text{Min}} \)) then for some (all) successors \( v' \) it holds that \( \Psi_X(v) \geq w(vv') + \Psi_X(v) \). This is the same as saying that Max can ensure that a non-negative weight is immediately seen from \( v \) in the \( \Psi_X \)-modified game, that is \( v \notin N_{\Phi_{\Psi_X}} \).
A.3 Correctness and complexity of Algorithms 2 and 3

We now prove Theorem 11.

**Theorem 11 (Adapted from [24,33,2]).** Algorithms 2 and 3 both compute $E_{U}^{+}$, and both can be implemented to run in $O(m + n \log n)$ operations.

**Proof.** We prove correctness of both algorithms using a similar induction, stating that $\Phi$ coincides with $E_{U}^{+}$ over $F^{c}$. In both cases this is true when the while loop starts, since $E_{U}^{+}$ is zero over $N_{G}$, so we focus on the inductions step. The two proofs below very are based on similar ideas, we separate them for clarity.

**Induction step for Algorithm 2.** There are two cases.

- If $F_{\text{Max}}^{\infty} \neq \emptyset$. Let $v \in F_{\text{Max}}^{\infty}$ be chosen by the algorithm. Since $esc_{F}(v) < \infty$, all positive edges outgoing from $v$ lead to $F^{c}$. Now an optimal $E_{U}^{+}$ strategy from $v$ should surely start with a positive edge, say, going to $v' \in F^{c}$. This concludes, since by induction, $E_{U}^{+}$ and $\Phi$ coincide over $F^{c}$.

- If $F_{\text{Max}}^{\infty} = \emptyset$. Then let $v \in F_{\text{Min}}^{\infty}$ be chosen by the algorithm, and $v'$ be such that $m = w(vv') + \Phi(v')$. We claim that the edge $vv'$ is $E_{U}^{+}$-optimal, which proves the wanted result by induction. Indeed, an other edge $vv''$ that ends in $F^{c}$ would lead to value $w(vv'') + \Phi(v'') \geq m$ by minimality. Now if Min plays an edge towards $F$, Max can force the game to remain in $F$ while visiting only non-negative edges (since $F_{\text{Max}}^{\infty} = \emptyset$). Therefore such a play remains in $F$ until potentially going to $F^{c}$ via a Min vertex, and thus its value is $\geq m$ by induction.

**Induction step for Algorithm 3.** Let $v$ be the vertex chosen by the algorithm, meaning $esc_{F}(v)$ is minimal, and in particular it is finite. Let $G'$ be the modified game at this stage of the algorithm, note that $G' = G_{\Phi}$.

- If $v \in V_{\text{Max}}$. Since $esc_{F}(v) < \infty$, all positive edges in $G'$ outgoing from $v$ lead to $F^{c}$. But since $\Phi(v) = 0$, positive edges outgoing from $v$ in $G'$ are also positive in $G$. Now an optimal $E_{U}^{+}$ strategy from $v$ in $G$ should surely start with a positive edge. The modified weights in $G'$ of edges from $v$ to $v' \in F^{c}$ are of the form $w(vv') + \Phi(v') = w(vv') + E_{U}^{+}(v')$ by induction. We conclude that the edge $vv'$ maximising $esc_{F}(v)$ satisfies $esc_{F}(vv') = E_{U}^{+}(v)$ which is also the final value of $\Phi(v)$.

- If $v \in V_{\text{Min}}$. Then let $v'$ be such that $esc_{F}(v) = w_{\Phi}(vv')$. We claim that the edge $vv'$ is $E_{U}^{+}$-optimal, which proves the wanted result by induction; starting with edge $vv'$ then playing optimally gives value $w(vv') + E_{U}^{+}(v')$, which correspond to $w_{\Phi}(vv') = esc_{F}(v)$ by induction. First, an other edge $vv''$ that ends in $F^{c}$ would lead to value $esc_{F}(vv'')$ by the same argument, which is $\geq esc_{F}(vv')$ (hence, less optimal for Min) by minimality. Now if Min plays an edge towards $F$, Max can force the game to either remain in $F$ while visiting only non-negative edges in $G'$ (since $F_{\text{Max}}^{\infty} = \emptyset$), or leave towards $F^{c}$ via an edge $uu'$ with modified weight $esc_{F}(uu') \geq esc_{F}(vv')$. But since $\Phi$ is 0 over $F$, non-negative edges in $G'$ are also non-negative in $G$, which concludes.
In both cases, the while loop terminates when $F^<\infty = \emptyset$. This means that Max can ensure that plays starting in $F$ visit only $\geq 0$ vertices. But since $\mathcal{G}$ is simple, this implies that $\text{En}^+$ is indeed $\infty$ over $F$, and thus it coincides with $\Phi$ everywhere.

Updating values of $F^<\infty$ requires, as is standard in such game algorithms, storing the number of positive edges, from each Max vertices towards $F$, and updating predecessors of vertices added to $F$. This incurs a runtime of $O(m)$.

For algorithm 3, updating minimal value of $\text{esc}_F$ requires using a priority queue, and the same technique can be applied in algorithm 2 to maintain the value of $m$. This induces a runtime of $O(n \log n)$, just like in Dijkstra’s algorithm [16]. ⊓ ⊔

A.4 Soundness and completeness of $\Psi_{\text{DPPI}}$

Lemma 12. The potential assigner $\Psi_{\text{DPPI}}$ is sound and complete for simple games.

Proof. Fix a simple game $\mathcal{G}$. We start proving soundness ($\Phi_{\text{DPPI}}(\mathcal{G}) \leq \text{En}_{\mathcal{G}}$). Let $\mathcal{G}_j, F_j, v_j$ and $\varphi_j$ be, respectively, the values of $\mathcal{G}, F, v$ and $\varphi$ after line 5 at the $j$th iteration of the algorithm, and let $\Psi_j = \varphi_j + \cdots + \varphi_1$, so that $\mathcal{G}_j = \mathcal{G}_{\Psi_j}$.

Thanks to compositionality (Corollary 9), it suffices to prove that $\varphi_j$ is sound in $\mathcal{G}_j$, so we should prove that $\varphi_j(v_j) = \text{esc}_{F_j}(v) \leq \text{En}_{\mathcal{G}_j}(v_j)$.

Note that $\Phi_j$ is 0 over $F_j$ so edges outgoing from vertices in $F_j$ have a weight in $\mathcal{G}_j$ greater or equal to their weight in $\mathcal{G}$. In particular, $F_j \subseteq V \setminus N_{\mathcal{G}_j}$, hence Max can ensure that only edges with non-negative weights are seen over $F_j$. Consider the following Max-strategy defined over $F_j$: if there is a non-negative edge towards $F_j$ from the current vertex $v$, play it; otherwise play an edge maximising $\text{esc}_{F_j}(v)$. We claim that this strategy achieves $\text{En}$-value $\geq \text{esc}_{F_j}(v)$ in $\mathcal{G}_j$. Consider a play $\pi$ from $v_j$ consistent with the strategy; there are two cases.

- If $\pi$ remains in $F_j$, then only non-negative weights are seen, and therefore since the game is simple, the value of the play is $\infty \geq \text{esc}_{F_j}(v)$.
- Otherwise, $\pi$ visits only non-negative edges within $F_j$ until following an edge $F_j \ni v \to v' \notin F_j$. Then the weight of this edge in $\mathcal{G}_j$ is $\geq \text{esc}_{F_j}(v)$, which concludes.

Finally, note that after the execution of the while-loop (line 9) $F = V \setminus \text{Attr}_{\mathcal{G}}(N_{\mathcal{G}})$. Therefore, $\text{En}_{\mathcal{G}}(v) = \infty$ for all those vertices.

We prove completeness, that is $\text{En}_{\mathcal{G}} \neq 0 \Rightarrow \Phi_{\text{DPPI}}(\mathcal{G}) > 0$. If $\text{En}_{\mathcal{G}} > 0$, then there is some vertex $v$ from which Max can immediately see a positive weight; note that $\text{esc}_{F}(v) > 0$ for all $F \subseteq V$ containing this vertex. Let $v_j$ correspond to the first such vertex encountered by the algorithm. Note that $v_j \notin N_{\mathcal{G}}$ and since $\Phi_j = 0$ we have $\text{esc}_{F}(v_j) = \text{esc}_{F}(v_j) > 0$. Therefore $\varphi_j(v_j) > 0$ hence $\Psi(\mathcal{G}) > 0$. ⊓ ⊔
B Comparisons between related algorithms

B.1 Comparison of potential assigners

Lemma 13. For every game $\mathcal{G}$,

$$\Psi^{+}_{\text{First}}(\mathcal{G}) \leq \Psi^{+}_{\text{En}}(\mathcal{G}) \leq \Psi^{+}_{\text{DPPI}}(\mathcal{G}), \text{ and } \Psi^{+}_{\text{GKK}}(\mathcal{G}) \leq \Psi^{+}_{\text{En}}(\mathcal{G}).$$

Moreover, there are games making these inequalities strict. The potential assigners $\Psi^{+}_{\text{First}}$ and $\Psi^{+}_{\text{GKK}}$ are incomparable.

Proof. We focus on the proof of the inequalities, and discuss below examples separating the different potentials.

$(\Psi^{+}_{\text{First}}(\mathcal{G}) \leq \Psi^{+}_{\text{En}}(\mathcal{G}))$ Follows directly from the fact that First$^{+} \leq$ En$^{+}$ over sequences of weights.

$(\Psi^{+}_{\text{GKK}}(\mathcal{G}) \leq \Psi^{+}_{\text{En}}(\mathcal{G}))$ By definition of $\Psi^{+}_{\text{GKK}}$, we have $\Psi^{+}_{\text{GKK}}(\mathcal{G}) \leq w_{+} \leq \text{En}_{w}^{+}$.

$(\Psi^{+}_{\text{En}}(\mathcal{G}) \leq \Psi^{+}_{\text{DPPI}}(\mathcal{G}))$ Let $\Phi = \Psi^{+}_{\text{DPPI}}(\mathcal{G})$. Let $v$ be a vertex assigned potential $\Phi(v) = x < \infty$ in the $j$th iteration of Algorithm 4 (the property trivially holds for $v$ with $\Phi(v) = \infty$). Assume by induction that $\text{En}_{w}^{+}(v') \leq \Phi(v')$ for all $v'$ that have been treated previously. Let $v' \notin F$ such that $vv' \in E$ is the transition determining $\Phi(v)$, that is, $\Phi(v) = w(vv') + \Phi(v')$. We distinguish two cases according to the player controlling $v$. If $v \in V_{\text{Min}}$, then:

$$\text{En}_{w}^{+}(v) \leq w(vv') + \text{En}_{w}^{+}(v') \leq w(vv') + \varphi(v') = \varphi(v),$$

where the second inequality follows by induction hypothesis.

If $v \in V_{\text{Max}}$, let $u$ be a successor of $v$ such that $\text{En}_{w}^{+}(v) = w(vu) + \text{En}_{w}^{+}(u)$. Note that $u \notin F$, as otherwise $\varphi(v) \geq \text{esc}_{F}(vu) = \infty$. Therefore:

$$\text{En}_{w}^{+}(v) = w(vu) + \text{En}_{w}^{+}(u) \leq w(vu) + \varphi(u) \leq w(vv') + \varphi(v') = \varphi(v),$$

where the second inequality follows by induction, and the third one because $vv'$ is the edge maximizing the escape weight from $F$ in $\mathcal{G}$. \hfill $\square$

A game separating PPI and DPPI was given in Figure 3.

Example 26 ($\Psi^{+}_{\text{First}} < \Psi^{+}_{\text{GKK}}$). Consider the game with a single vertex $v$ and two self loops, with weights 1 and $W$. We have that $\Psi^{+}_{\text{First}}(\mathcal{G})(v) = 1$, and SVI takes $W$ iterations to realize that $\text{En}_{w}(v) = \infty$. However, $\Psi^{+}_{\text{GKK}}(\mathcal{G})(v) = \Psi^{+}_{\text{En}}(\mathcal{G})(v)\Psi^{+}_{\text{DPPI}}(\mathcal{G})(v) = \infty$; all the other algorithms terminate in a single iteration.

Example 27 ($\Psi^{+}_{\text{GKK}} < \Psi^{+}_{\text{First}}$ is slow). We note that for all game $\mathcal{G}$, the image of $\Psi^{+}_{\text{GKK}}(\mathcal{G})$ contains at most two values: 0 and $w_{+}$ (or $w_{-}$). Consider the game with three vertices controlled by Max $V = V_{\text{Max}} = \{v_{1}, v_{2}, v_{N}\}$, and edges given by: $v_{N} \xrightarrow{1} v_{N}, v_{1} \xrightarrow{1} v_{N}, v_{2} \xrightarrow{2} v_{N}. \text{ Then, we have } w_{+} = w_{-} = 1, \text{ and } \Psi^{+}_{\text{GKK}}(v_{1}) = \Psi^{+}_{\text{GKK}}(v_{2}) = 1$. The GKK algorithm takes 2 iterations to solve this game. However, $\Psi^{+}_{\text{First}}(v_{2}) = 2$, and SVI takes a single iteration to solve the game.
B.2 Comparison between PPI and OSI

We now describe the algorithm OSI, explaining the similarities and differences with PPI, our presentation within the fast value iteration framework.\footnote{Note that OSI was originally presented exclusively over parity games. It can easily be generalized to energy games, in the following we will always refer to this straightforward generalization.} \footnote{We note that Player 0 in \cite{33,27} corresponds to our player Max.}

OSI relies on the notion of estimations, which correspond to our potentials. To update an estimation (basic update in \cite[p.377]{33}), OSI uses the auxiliary update game $E_\phi$, obtained from $G_\phi$ by: i) adding a sink state $\perp$ to which Max can retreat at any point, ii) Max-choices are restricted to non-negative edges. In this game, Max tries to maximize the weight of a play before reaching $\perp$. That value almost coincides with the $E^+$-values of $G_\phi$, and the subroutine used to solve the update game is very similar to the Algorithm 2; the main difference is that in $E^+$ we stop the game as soon as one of the players produces a negative edge. Due to this difference, some extra technical steps are required in the presentation of OSI:

- The presentation of the algorithm is restricted to bipartite graphs.
- An initialisation step in which a first potential $\phi_0: V \to \mathbb{N}$ is computed is required. In the case of a bipartite graph, these are just the First$^+$-values of Min-vertices.
- Before each basic update state, we need to ensure that Min will not have the opportunity to visit negative edges in the update game. To this end, the current potential $\phi$ needs to be decreased in some Max-positions (point 2 at the bottom of \cite[p.379]{33}).

It is worth mentioning that, soon after the introduction of Schewe’s algorithm, Luttenberger \cite{27} proposed a reformulation as an explicit switching policy in the strategy improvement framework. Although a potential is still used to guide the updates of the strategies, it comes organically as the evaluation of the current strategies. To compute this evaluation, Luttenberger uses an adaptation of the Bellman-Ford algorithm, which is less efficient than Dijkstra’s.

For the reasons stated above, we see PPI as a polished and streamlined version of Schewe’s algorithm. In particular, PPI avoids the introduction of an additional sink vertex, answering a question by Björklund and Vorobiov \cite[Conclusion]{4}. The discrepancies on the running time (see Section 5) can be explained by (1) the extra computation steps that appear in the original description of OSI, and (2) the initialization to a slightly different potential in OSI’s first step.

B.3 Comparison between QDPM and PPI

We now describe the algorithm QDPM, explaining the similarities and differences with PPI, our presentation within the fast value iteration framework.

The presentation of QDPM from \cite{2} is based on the notion of quasi dominions. A subset of positions $Q \subseteq V$ is a quasi dominion if player Max has a strategy
ensuring to visit only non-negative weight as long as the play does not exit $Q$. Therefore, player Min has an incentive to leave such a region as soon as possible. We observe that the set $F = N_G^c$ from which Min cannot force to immediately see a negative edge is a quasi dominion in the game $G$. The algorithm PPI finds a strategy for Min to leave this quasi dominion minimising the energy.

The main iteration principle of QDPM is provided by the operator $\text{prg}_+ ([2, Alg. 1])$. This almost corresponds to Algorithm 3 in our presentation. That is why we consider that both algorithms use the same underlying mechanism. However, there are some differences between QDPM and PPI that may lead to different executions over the same game:

- QDPM does not apply potential updates modifying the game. Instead, it carries the information in a potential $\mu$ (progress measure in the terminology of [2]), which is updated in each iteration. The information carried by the potential $\mu$ is used in the other iterations by the algorithm. (By itself, this does not provoke differences in executions.)
- QDPM does not initialise the quasi dominion $F$ to $N_G^c$. Instead, $F$ is the set of positions which are assigned value $> 0$ by the potential $\mu$ coming from previous iterations ($F = \mu^{-1}(0)$). In order to enlarge this set, a first small update (corresponding to a potential update of $\Psi_{\text{First}+}$) is applied to $F$, this corresponds to $\text{prg}_0(\mu^{-1}(0))$ in [2, p.7]).
- It is important to notice that due to this first initialization step applying a first potential $\text{prg}_0$, the games treated by PPI and QDPM slightly differ. Over several iterations, the behaviour of both algorithms may therefore diverge. We observe empirically that while there may be rare differences between the number of iterations of the two algorithms, they remain negligible compared to the total number, and are not biased towards one or the other algorithm.
- Importantly, QDPM includes some smart implementation techniques to avoid considering all vertices in the computation of $\text{prg}_0$ and $\text{prg}_+$ [2, Sect. 5], improving their theoretical complexity upper bound to match the one of [6].

### B.4 Implementation differences comparison between state-weighted and vertex-weighted games

We now propose an explanation why QDPM, as implemented by [2], performing an order of magnitude quicker than our implementation of PPI, basing on the fact that the QDPM is based on vertex-weighted games whereas PPI is based on edge-weighted ones. Recall that we study games where weights are exponential (this is also the case when translating from parity games with linearly many priorities); therefore essentially the whole runtime is spent on performing such operations, which are either additions or comparisons.

These are broken into three categories:

1. Updating the total potential $\Phi$ of each vertex. This requires roughly $n$ additions per iteration.
2. Insertions in priority queues. This requires roughly $n \log n$ comparisons.
3. Weight comparisons. This is where the difference lies. Here, we should compare the modified weights of two outgoing edges \( vv_1 \) and \( vv_2 \) from a given vertex \( v \). In the edge-weighted scenario, this amounts to comparing \(-\Phi(v) + \Phi(v_1) + w(vv_1)\) with \(-\Phi(v) + \Phi(v_2) + w(vv_2)\), or equivalently \(\Phi(v_1) + w(vv_1)\) with \(\Phi(v_2) + w(vv_2)\). This requires 2 additions and 1 comparison, which amounts overall to roughly \(2m\) additions and \(m\) comparisons per iteration.

In contrast, in the vertex-weighted scenario, \(w(vv_1) = w(vv_2)\), so it is enough to compare \(\Phi(v_1)\) with \(\Phi(v_2)\), leading to \(m\) comparisons, which saves on \(2m\) costly additions per iteration.

In total, we get the following numbers:

| type             | comparisons | additions |
|------------------|-------------|-----------|
| edge-weighted    | \(n \log n\) | \(n + 2m\) |
| vertex-weighted  | \(n \log n\) | \(n\)     |

To give concrete estimates, we have compared runtimes between additions and comparisons (in the GMP libraries), for weights corresponding to the our biggest instances, reporting a ratio of over 4 orders of magnitude \((10^4)\). For sparse games \((m = 2n)\) this explains a factor of roughly 5 between the two implementations, which is more that the difference in runtimes. For dense games, this gives a linear factor in \(n\) on the number of additions (although arguably dense games typically have smaller weights).

C  Friedmann’s family of examples

We include the performance (on number of iterations) of our algorithms against the family of examples proposed by Friedmann [17] (Figure 6). We also ran the same experiments in the randomized setting, where instances are first perturbed by random potentials sampled according to a normal distribution (Figure 7).

It is not surprising that PPI-alt performs a constant (namely, 2) number of iterations, because the dual algorithm immediately attracts the whole game to the single negative cycle (the instances are by no means designed to be resilient to such dual algorithms). We indeed observe that QDPM and PPI perform a linear number of iterations, as claimed in the conclusion. Remarkably, DPPI performs a constant number of iterations on roughly half of the instances, while on a few instances it performs slightly more iterations than PPI (this is not a contradiction to Theorem 13 as explained just above it).

In the randomized setting, we observe some speedup (for PPI) on some of the instances, which shows that it could make sense to run (in parallel) the algorithm on perturbated inputs. However, we remark that in most of the cases the number of iterations is noticeably increased. This constitutes by no means a serious experimental study of this phenomenon, which we leave to future work. In particular, it would be more meaningful to run this experiment on instances requiring super linear number of updates (which are not available at the moment).
Fig. 6. Number of iterations against Friedmann’s hard examples.

Fig. 7. Number of iterations against Friedmann’s hard examples in the randomized setting with initial perturbations.