Value Iteration Using Universal Graphs and the Complexity of Mean Payoff Games

Nathanaël Fijalkow, Pawel Gawrychowski, Pierre Ohlmann

Value Iteration Using Universal Graphs and the Complexity of Mean Payoff Games

Nathanaël Fijalkow
CNRS, LaBRI, Bordeaux, France
The Alan Turing Institute of data science, London, UK

Paweł Gawrychowski
Institute of Computer Science, University of Wrocław, Poland

Pierre Ohlmann
Université de Paris, IRIF, CNRS, France

Abstract

We study the computational complexity of solving mean payoff games. This class of games can be seen as an extension of parity games, and they have similar complexity status: in both cases solving them is in $\mathbf{NP} \cap \mathbf{coNP}$ and not known to be in $\mathbf{P}$. In a breakthrough result Calude, Jain, Khoussainov, Li, and Stephan constructed in 2017 a quasipolynomial time algorithm for solving parity games, which was quickly followed by a few other algorithms with the same complexity. Our objective is to investigate how these techniques can be extended to mean payoff games.

The starting point is the combinatorial notion of universal trees: all quasipolynomial time algorithms for parity games have been shown to exploit universal trees. Universal graphs extend universal trees to arbitrary (positionally determined) objectives. We show that they yield a family of value iteration algorithms for solving mean payoff games which includes the value iteration algorithm due to Brim, Chaloupka, Doyen, Gentilini, and Raskin.

The contribution of this paper is to prove tight bounds on the complexity of algorithms for mean payoff games using universal graphs. We consider two parameters: the largest weight N in absolute value and the number k of weights. The dependence in N in the existing value iteration algorithm is linear, we show that this can be improved to $N^{1-1/n}$ and obtain a matching lower bound. However, we show that we cannot break the linear dependence in the exponent in the number k of weights implying that universal graphs do not yield a quasipolynomial time algorithm for solving mean payoff games.

2012 ACM Subject Classification Theory of computation → Automata over infinite objects; Theory of computation → Algorithmic game theory and mechanism design

Keywords and phrases Mean payoff games, Universal graphs, Value iteration

Digital Object Identifier 10.4230/LIPIcs.MFCS.2020.34

Related Version A full version of the paper is available on arXiv [14] at `https://arxiv.org/abs/1812.07072`.

Funding The first and third authors were (partially) funded by the DeLTA project (ANR-16-CE40-0007).

1 Introduction

A mean payoff game is played over a finite graph whose edges are labelled by integer weights. The interaction of the two players, called Eve and Adam, describe a path in the graph. The goal of Eve is to ensure that the (infimum) limit of the weights average is non-negative.

The model of mean payoff games was introduced independently by Ehrenfeucht and Mycielski [11] and by Gurvich, Karzanov, and Khachiyan [18]. A fundamental property proved in both papers is that such games are positionally determined, meaning that for both players, if there exists a strategy ensuring mean payoff, then there exists one using no

memory at all. This holds for both players and is the key argument implying the intriguing complexity status of solving mean payoff games: the decision problem is in **NP** and in **coNP**, but not known to be solvable in polynomial time. It is unlikely to be **NP**-complete, since this would imply that **NP** = **coNP**. Hence such a problem is either solvable in polynomial time or an interesting piece in the landscape of computational complexity. This is one of the reasons that makes the study of mean payoff games exciting. In addition to game theory, the study of mean payoff games is motivated by verification and synthesis of programs, and by their intricate connections to optimisation and linear programming. For instance, the model of mean payoff games has been recently shown to be connected to the existence of a strongly polynomial time algorithm for linear programming, which is the 9^{th} item in Smale's list of problems for the 21^{st} century [30]. Allamigeon, Benchimol, Gaubert, and Joswig [1] have shown that a strongly polynomial time semi-algebraic pivoting rule in linear programming would solve mean payoff games in strongly polynomial time.

The seminal paper of Zwick and Paterson [32] relates mean payoff games to discounted payoff games and simple stochastic games, and most relevant to our work, constructs an algorithm for solving mean payoff games with complexity $O(n^2mN)$, where n is the number of vertices, m the number of edges, and N the largest weight in absolute value. If the weights are given in unary N is polynomial in the representation, so we say that the algorithm is pseudopolynomial. The question whether there exists a polynomial time algorithm for mean payoff games with the usual representation of weights, meaning in binary, is open. The currently fastest algorithm for mean payoff games is randomised and achieves subexponential complexity $2^{\tilde{O}(\sqrt{n})}$. It is based on randomised pivoting rules for the simplex algorithm devised by Kalai [21, 22] and Matoušek, Sharir and Welzl [27].

We are in this work interested in deterministic algorithms for solving mean payoff games. There are two fastest deterministic algorithms: the value iteration algorithm of Brim, Chaloupka, Doyen, Gentilini, and Raskin [3], which has complexity $O(nmN)$, and the algorithm of Lifshits and Pavlov [26] with complexity $O(nm2^n)$. They are incomparable: the former is better when $N \leq 2^n$ and otherwise the latter prevails. Very recently Dorfman, Kaplan, and Zwick [10] presented an improved version of the value iteration algorithm with a complexity $O(\min(nmN, nm2^{n/2}\log(N)))$, an improvement over the previous algorithm when $N = \Omega(n2^{n/2})$.

Solving a mean payoff game is very related to constructing an optimal strategy, meaning one achieving the highest possible value. The state of the art for this problem is due to Comin and Rizzi [7] who designed a pseudopolynomial time algorithm.

Parity games

It is most instructive in this context to think of parity games as a subclass of mean payoff games. Indeed, a parity game with priorities in $0, \ldots, d$ is turned into an equivalent mean payoff game by replacing an edge of priority p by one of weight $(-n)^p$. The breakthrough result of Calude, Jain, Khoussainov, Li, and Stephan [4] was to construct a quasipolynomial time algorithm for solving parity games. Following decades of exponential and subexponential algorithms, this very surprising result triggered further research: soon after further quasipolynomial time algorithms were constructed reporting almost the same complexity, which is roughly $n^{O(\log d)}$. Let us classify them in two families.

The first family of algorithms includes the original algorithm by Calude, Jain, Khoussainov, Li, and Stephan [4] (see also [12] for a presentation of the algorithm as value iteration), then the succinct progress measure algorithm by Jurdziński and Lazić [19] (see also [13] for a presentation of the algorithm using universal trees explicitly) and the register games algorithm

by Lehtinen [24] (see also [29] for a presentation of the algorithm using good-for-small-games automata explicitly). Bojańczyk and Czerwiński [2] introduced the separation question, describing a family of algorithms for solving parity games based on *separating automata*, and showed that the first quasipolynomial time algorithm yields a quasipolynomial solution to the separation question. Later Czerwiński, Daviaud, Fijalkow, Jurdziński, Lazić, and Parys [8] showed that the other two algorithms also yield quasipolynomial solutions to the separation question. The main contribution of [8] is to show that any separating automaton contains a *universal tree* in its set of states; in other words, the three algorithms in this first family induce three (different) constructions of universal trees.

The second family of algorithms is so-called Zielonka-type algorithms, inspired by the exponential time algorithm by Zielonka [31]. The first quasipolynomial time algorithm is due to Parys [28], its complexity was improved by Lehtinen, Schewe, and Wojtczak [25]. Recently Jurdziński and Morvan [20] constructed a universal attractor decomposition algorithm encompassing all three algorithms: each algorithm is parameterised by the choice of two universal trees (one of each player).

From universal trees to universal graphs

All quasipolynomial time algorithms for parity games fall in one of two families, and both are based on the combinatorial notion of universal trees. This notion is by now well understood, with almost matching upper and lower bounds on the size of universal trees [13, 8]. The lower bound implies a complexity barrier applying to both families of algorithms, hence to all known quasipolynomial time algorithms.

Universal trees arise in the study of the parity objective, the tree structure representing the nested behaviour of priorities. Colcombet and Fijalkow [5, 6] introduced the notion of *universal graphs* to extend universal trees from parity objectives to arbitrary (positionally determined) objectives. The main result of [6] is an equivalence result between *good-for-small-games automata*[1] and *universal graphs*. More specifically, a good-for-small-games automaton induces a universal graph of the same size, and vice versa. This equivalence extends the results of [8] relating separating automata and universal trees to any positionally determined objectives, so in particular for parity and mean payoff games.

Universal graphs for labelling schemes

The notion of universal graphs has been extensively studied in the unrelated context of labelling schemes: the goal is to assign a short bitstring (called a label) to every node of a graph so that a query concerning two nodes can be answered by looking at their corresponding labels alone. As a prime example, labelling nodes of an undirected graph for adjacency queries is known to be equivalent to constructing a so-called subgraph induced universal graph [23]. Even though for some queries approaches based on an appropriately chosen notion of universal graphs are known to be suboptimal [16], we also have examples in which they allow for a significantly simpler and more efficient solution [17].

[1] Good-for-small-games automata extend the notion of separating automata by including restricted non-determinism. This notion is useful to capture Lehtinen's register games algorithm, as explained in [6], see also [29] for a similar point of view.

Contributions

Section 2 is devoted to a development of the theory of universal graphs for solving games with arbitrary positionally determined objectives: we show that universal graphs yield a family of *value iteration algorithms* whose time complexity is proportional to the number of vertices of the universal graph and space complexity proportional to its logarithm. This improves upon the previous approaches [2, 6] by reduction to safety games whose (time and space) complexity is proportional either to the number of edges or to the number of vertices of the universal graph.

Section 3 applies this result for mean payoff objectives implying that the value iteration algorithm due to Brim, Chaloupka, Doyen, Gentilini, and Raskin [3] is an instance of the family of algorithms based on universal graphs. The rest of the paper gives upper and lower bounds on the size of universal graphs for mean payoff games along two parameters: the largest weight N in absolute value and the number k of distinct weights.

Section 4: Universal graphs parametrised by the largest weight

The universal graph underlying the value iteration algorithm of [3] has size nN. We show that this can be improved by constructing a universal graph of size at most $2n^2N^{1-1/n}$, which is asymptotically smaller than nN when N is exponential in n. This induces a new algorithm with time complexity $O(nm(nN)^{1-1/n})$ and space complexity $O(n\log(N))$. We then prove a matching lower bound: all universal graphs have size at least $N^{1-1/n}$. This implies that the linear dependence in N cannot be significantly improved for algorithms using universal graphs.

Section 5: Universal graphs parametrised by the number k of different weights

There is a universal graph of size $O(n^k)$ for solving mean payoff games of size n with at most k weights. We prove an almost matching lower bound: for all k, there exists a set of k weights such that all corresponding universal graphs have size at least $\Omega(n^{k-2})$. This implies that algorithms using universal graphs cannot break the $O(n^{\Omega(k)})$ barrier, and in particular do not have quasipolynomial complexity.

2 Universal graphs and value iteration algorithms

In this section we define universal graphs and show how they yield algorithms for solving games. There are two existing approaches both constructing reductions to safety games. The most direct one [6] yields algorithms whose (time and space) complexity is proportional to the number of edges of the universal graph. A better complexity is obtained by relating universal graphs to separating automata as introduced in [2] and yields algorithms whose (time and space) complexity is proportional to the number of vertices of the universal graph.

The core of this section is to show that universal graphs also yield a family of value iteration algorithms, whose time complexity is proportional to the number of vertices of the universal graph. The main benefit is in the space complexity, which becomes proportional to the logarithm of the number of vertices of the universal graph. The developments of this section are very general as they apply to any objective which is positionally determined. We will instantiate this to the class of mean payoff objectives in the next section.

We write $[i, j]$ for the interval $\{i, i+1, \ldots, j-1, j\}$, and use parentheses to exclude extremal values, so $[i, j)$ is $\{i, i+1, \ldots, j-1\}$. We let C denote a set of colours and write C^* for finite sequences of colours (also called finite words), C^+ for finite non-empty sequences, and C^ω for infinite sequences (also called infinite words).

Universal graphs

We deliberately postpone the definitions related to games: indeed the notion of universal graphs is a purely combinatorial notion on graphs and ignores the interactive aspects of games.

Graphs

We consider edge labelled directed graphs: a graph is given by a (finite) set V of vertices and a (finite) set $E \subseteq V \times C \times V$ of edges. A vertex v for which there exists no outgoing edge $(v, \ell, v') \in E$ is called a sink. We let n denote the number of vertices and m the number of edges. The size of a graph is its number of vertices.

Homomorphisms

For two graphs G, G', a homomorphism $\phi : G \to G'$ maps the vertices of G to the vertices of G' such that

$$(v, \ell, v') \in E \implies (\phi(v), \ell, \phi(v')) \in E'.$$

As a simple example that will be useful later on, note that if G' is a super graph of G, meaning they have the same set of vertices and every edge in G is also in G', then the identity is a homomorphism $G \to G'$. We say that G maps into G' if there exists a homomorphism $G \to G'$.

Paths

A path π is a (finite or infinite) sequence of consecutive edges, where consecutive means that the third component of a triple in the sequence matches the first component of the next triple. In the case of a finite path we write $\text{last}(\pi)$ for the last vertex in π. We write $\pi = (v_0, \ell_0, v_1)(v_1, \ell_1, v_2) \cdots$ and let $\pi_{\leq i}$ denote the prefix of π of length i, meaning $\pi_{\leq i} = (v_0, \ell_0, v_1) \cdots (v_{i-1}, \ell_{i-1}, v_i)$.

Objectives

An objective is a set $\Omega \subseteq C^\omega$ of infinite sequences of colours. A sequence of colours belonging to Ω is said to satisfy Ω. We say that a path in a graph satisfies Ω, or that it is winning when Ω is clear from context, if the sequence of labels it visits belongs to Ω. A path is said to be maximal if it is either infinite or ends in a sink.

In the remainder of this section we will work with a generic objective Ω. However it is convenient to assume that Ω is *prefix independent*. Formally, this means that $\Omega = C^* \cdot \Omega$, or equivalently for all $w \in C^*$ and $\rho \in C^\omega$, we have $\rho \in \Omega \iff w \cdot \rho \in \Omega$. This assumption can be lifted at the price of working with graphs with a distinguished initial vertex v_0 which must be preserved by homomorphisms. Although all results extend, we chose not to work in this slightly more general setting since the mean payoff objective is prefix independent.

▶ **Definition 1** (Graphs satisfying an objective). *Let Ω be a prefix independent objective. A graph satisfies Ω if all maximal paths are infinite and winning.*

Note that if a graph contains a sink, it does not satisfy Ω because it contains some finite maximal path.

▶ **Definition 2** (Universal graphs). *Let Ω be a prefix independent objective. A graph $\mathcal{U}$ is (n, Ω)-universal if it satisfies the following two properties:*

1. *it satisfies Ω,*
2. *for any graph G of size n satisfying Ω, there exists a homomorphism $\phi : G \to \mathcal{U}$.*

When considering a universal graph $\mathcal{U}$ we typically let $V_{\mathcal{U}}$ and $E_{\mathcal{U}}$ denote the sets of vertices and edges of $\mathcal{U}$, respectively.

It is not clear that for any objective Ω and $n \in \mathbb{N}$ there exists an (n, Ω)-universal graph. Indeed, the definition creates a tension between "satisfying Ω", which restricts the structure, and "mapping any graph of size n satisfying Ω", implying that the graph witnesses varied behaviours. A simple construction of an (n, Ω)-universal graph is to take the disjoint union of all graphs of size n satisfying Ω. Since up to renaming of vertices there are finitely many such graphs for a fixed n, this yields a very large but finite (n, Ω)-universal graph.

Solving games using universal graphs by reduction to safety games

We now introduce infinite duration games on graphs and describe two first approaches for solving games using universal graphs by reduction to safety games [2, 6]. We start with the most direct one [6] as it does not require any deeper understanding of universal graphs, and then briefly discuss the second one which actually relies on their connections to separating automata [2].

Arenas

An arena is given by a graph *containing no sink* together with a partition $\mathrm{V}_{\mathrm{Eve}} \uplus \mathrm{V}_{\mathrm{Adam}}$ of its set V of vertices describing which player controls each vertex.

Games

A game is given by an arena and an objective. We often let $\mathcal{G}$ denote a game, its size is the size of the underlying graph. It is played as follows. A token is placed on some initial vertex v_{init}, and the player who controls this vertex pushes the token along an edge, reaching a new vertex; the player who controls this new vertex takes over, and this interaction goes on forever describing an infinite path.

Strategies

We write $\mathrm{Path}_{\mathcal{G}}^{\mathrm{Eve}}(v_{\mathrm{init}})$ for the set of finite paths of $\mathcal{G}$ starting from v_{init} and ending in $\mathrm{V}_{\mathrm{Eve}}$. A strategy for Eve from v_{init} is a map $\sigma : \mathrm{Path}_{\mathcal{G}}^{\mathrm{Eve}}(v_{\mathrm{init}}) \to E$ such that for all $\pi \in \mathrm{Path}_{\mathcal{G}}^{\mathrm{Eve}}(v_{\mathrm{init}})$, $\sigma(\pi)$ is an edge in E from $\mathrm{last}(\pi)$. Note that we always take the point of view of Eve, so a strategy implicitely means a strategy of Eve, and winning means winning for Eve. We say that an infinite path $\pi = (v_0, \ell_0, v_1)(v_1, \ell_1, v_2) \cdots$ is consistent with the strategy σ if for all i, if $v_i \in \mathrm{V}_{\mathrm{Eve}}$, then $\sigma(\pi_{\leq i}) = (v_i, \ell_i, v_{i+1})$.

A strategy σ is winning from v_{init} if all infinite paths starting from v_{init} and consistent with σ are winning. Solving a game is the following decision problem:

INPUT: a game $\mathcal{G}$ and an initial vertex v_{init}

OUTPUT: "yes" if Eve has a winning strategy from v_{init}, "no" otherwise.

We say that v_{init} is a winning vertex of $\mathcal{G}$ when the answer to the above problem is positive.

Positional strategies

Positional strategies make decisions only considering the current vertex: $\sigma : \mathrm{V}_{\mathrm{Eve}} \to E$. A positional strategy induces a strategy $\widehat{\sigma} : \mathrm{Path}_{\mathcal{G}}^{\mathrm{Eve}}(v_{\mathrm{init}}) \to E$ by $\widehat{\sigma}(\pi) = \sigma(\mathrm{last}(\pi))$, where by convention the last vertex of the empty path is the initial vertex v_{init}.

▶ **Definition 3** (Positionally determined objectives). *We say that an objective Ω is positionally determined if for every game with objective Ω and initial vertex v_{init}, whenever there exists a winning strategy from v_{init} then there exists a positional winning strategy from v_{init}.*

Given a game $\mathcal{G}$, an initial vertex v_{init}, and a positional strategy σ we let $\mathcal{G}[\sigma, v_{\mathrm{init}}]$ denote the graph obtained by restricting $\mathcal{G}$ to vertices reached by σ from v_{init} and to the moves prescribed by σ. Formally, the set of vertices and edges is

$$\begin{array}{rcl} V[\sigma, v_{\mathrm{init}}] & = & \{v \in V : \text{there exists a path from } v_{\mathrm{init}} \text{ to } v \text{ consistent with } \sigma\}, \\ E[\sigma, v_{\mathrm{init}}] & = & \{(v, \ell, v') \in E : v \in \mathrm{V}_{\mathrm{Adam}} \text{ or } (v \in \mathrm{V}_{\mathrm{Eve}} \text{ and } \sigma(v) = (v, \ell, v'))\} \\ & \cap & V[\sigma, v_{\mathrm{init}}] \times C \times V[\sigma, v_{\mathrm{init}}]. \end{array}$$

Our assumption that games do not contain sinks is essential for the following fact.

▶ **Fact 1.** *Let Ω be a prefix independent objective, $\mathcal{G}$ a game, v_{init} an initial vertex, and σ a positional strategy. Then the strategy σ is winning from v_{init} if and only if the graph $\mathcal{G}[\sigma, v_{init}]$ satisfies Ω.*

Safety games

The safety objective `Safe` on two colours $C = \{\varepsilon, \mathrm{Lose}\}$ is given by $\texttt{Safe} = \{\varepsilon^\omega\}$. In words, an infinite path is winning if it avoids the colour Lose. The following lemma is folklore.

▶ **Lemma 4.** *Given a safety game with m edges, there exists an algorithm running in time and space $O(m)$ which computes the set of winning vertices.*

Consider a game $\mathcal{G}$ with objective Ω of size n and an (n, Ω)-universal graph $\mathcal{U}$, we construct a safety game $\mathcal{G} \triangleright \mathcal{U}$ as follows. The arena for the game $\mathcal{G} \triangleright \mathcal{U}$ is given by the following set of vertices and edges

$$\begin{array}{rcl} \mathrm{V}'_{\mathrm{Eve}} & = & \mathrm{V}_{\mathrm{Eve}} \times V_{\mathcal{U}} \ \uplus \ V_{\mathcal{U}} \times E, \\ \mathrm{V}'_{\mathrm{Adam}} & = & \mathrm{V}_{\mathrm{Adam}} \times V_{\mathcal{U}}, \\ E' & = & \{((v, s), \varepsilon, (s, \ell, v')) : s \in V_{\mathcal{U}}, (v, \ell, v') \in E\} \\ & \cup & \{((s, \ell, v'), \varepsilon, (v', s')) : (s, \ell, s') \in E_{\mathcal{U}}, v' \in V\} \\ & \cup & \{((s, \ell, v'), \mathrm{Lose}, (s, \ell, v')) : s \in V_{\mathcal{U}}, (v, \ell, v') \in E\} \end{array}$$

In words: the game $\mathcal{G} \triangleright \mathcal{U}$ simulates $\mathcal{G}$. From (v, s), the following two steps occur. First, the player who controls v picks an edge $(v, \ell, v') \in E$ as he would in $\mathcal{G}$, and second, Eve chooses which edge (s, ℓ, s') to follow in the universal graph $\mathcal{U}$. If Eve is unable to play in $\mathcal{U}$ (because there are no outgoing edges of the form (s, ℓ, s')), she is forced to choose $((s, \ell, v'), \mathrm{Lose}, (s, \ell, v'))$ and lose the safety game.

▶ **Theorem 5.** *Let Ω be a prefix independent positionally determined objective. Let $\mathcal{G}$ be a game of size n with objective Ω and $\mathcal{U}$ an (n, Ω)-universal graph. Let v_{init} be a vertex. Then Eve has a winning strategy in $\mathcal{G}$ from v_{init} if and only if there exists a vertex s_{init} in $\mathcal{U}$ such that she has a winning strategy in the safety game $\mathcal{G} \triangleright \mathcal{U}$ from (v_{init}, s_{init}).*

This theorem can be used to reduce games with objective Ω to safety games, yielding an algorithm whose complexity is proportional to the number of edges of $\mathcal{U}$. Indeed, recall that the complexity of solving a safety game is proportional to the number of edges. The number of edges of $\mathcal{G} \triangleright \mathcal{U}$ is $O(n \cdot m_{\mathcal{U}})$ so the overall complexity is $O(n \cdot m_{\mathcal{U}})$.

Moving towards more efficient algorithms, we proceed to describing the structure of saturated universal graphs.

▶ **Definition 6** (Linear graphs). *A graph G is linear if there exists a total order $\leq$ on the vertices of G satisfying the following two properties:*

$$\text{if } v' \leq v \text{ and } (v', \ell, v'') \in E, \text{ then } (v, \ell, v'') \in E,$$
$$\text{if } (v, \ell, v') \in E \text{ and } v'' \leq v', \text{ then } (v, \ell, v'') \in E.$$

We refer to the first property as left composition and the second as right composition.

The following lemma can be seen as a consequence of a result from [6], which we adapt to our vocabulary.

▶ **Lemma 7** (from Lemma 9 in [6]). *Let Ω be a prefix independent positionally determined objective. Then for every graph G satisfying Ω, there exists a linear graph G' no larger than G and a homomorphism $\phi : G \to G'$.*

A direct consequence of Lemma 7 is if there exists an (n, Ω)-universal graph $\mathcal{U}$, then there exists an (n, Ω)-universal linear graph no larger than $\mathcal{U}$.

This is the key ingredient for determinising universal graphs [6]: an (n, Ω)-universal graph is deterministic if for every vertex $s \in V_{\mathcal{U}}$ and colour $\ell \in C$ there exists at most one edge $(s, \ell, s') \in E_{\mathcal{U}}$. We state the result but do not elaborate further.

▶ **Corollary 8.** *Let Ω be a prefix independent positionally determined objective and $\mathcal{U}$ an (n, Ω)-universal graph. There exists a deterministic (n, Ω)-universal graph no larger than $\mathcal{U}$.*

This yields an algorithm of improved time and space complexity $O(n|C| \cdot n_{\mathcal{U}})$ using the reduction to the safety game $\mathcal{G} \triangleright \mathcal{U}$.

Solving games using universal graphs by value iteration

We now further analyse the structure of universal graphs in order to define a notion of progress measures which in turn will allow us to construct value iteration algorithms. This improves on the two previous approaches by drastically decreasing the space complexity.

Let us consider a linear (n, Ω)-universal graph $\mathcal{U}$. We let $\mathcal{U}_\top$ denote the linear graph $\mathcal{U}$ extended with a vertex $\top$ and the edges $(\top, \ell, \top)$ for all $\ell \in C$, as well as $(\top, \ell, s)$ for $s \in V_{\mathcal{U}}$. We extend $\leq$ to $\mathcal{U}_\top$ by making $\top$ maximal, which makes $\mathcal{U}_\top$ linear. We define a notion of progress measures for games extending the notion of homomorphisms for graphs.

▶ **Definition 9** (Progress measures). *A progress measure for the game $\mathcal{G}$ is a function $\phi : V \to V_{\mathcal{U}_\top}$ such that:*

- *for $v \in V_{Eve}$, there exists $(v, \ell, v') \in E$ such that $(\phi(v), \ell, \phi(v')) \in E_{\mathcal{U}_\top}$,*
- *for $v \in V_{Adam}$, for all $(v, \ell, v') \in E$ we have $(\phi(v), \ell, \phi(v')) \in E_{\mathcal{U}_\top}$.*

We let Π denote the set of functions $\phi : V \to V_{\mathcal{U}_\top}$ and equip it with the pointwise order induced by $\leq$: we say that $\phi \leq \phi'$ if for all vertices $v \in V$ we have $\phi(v) \leq \phi'(v)$.

▶ **Theorem 10.** *Let Ω be a prefix independent positionally determined objective, $\mathcal{G}$ a game with objective Ω, and $\mathcal{U}$ an (n, Ω)-universal linear graph. There exists a unique minimal progress measure $\phi \in \Pi$ and it has the following property: for all vertices $v_{init} \in V$, we have $\phi(v_{init}) \neq \top$ if and only if Eve has a winning strategy from v_{init}.*

The value iteration algorithm constructs the minimal progress measure. The key insight is to approach this task as a smallest fixed point computation: progress measures are seen as the pre fixed points of a set of operators, and the algorithm computes the smallest fixed point for this set of operators, *i.e.* the minimal progress measure.

Let $\mathcal{U}$ be an (n, Ω)-universal linear graph. We let s_{init} the smallest vertex in $\mathcal{U}$. We let $\delta : V_{\mathcal{U}} \times C \to V_{\mathcal{U}}$ denote the function defined by $\delta(s', \ell) = \min\{s : (s, \ell, s') \in E_{\mathcal{U}}\}$. Note that $s \geq \delta(s', \ell)$ if and only if $(s, \ell, s') \in E_{\mathcal{U}}$ thanks to left composition in $\mathcal{U}$.

We introduce a set of operators $\mathbf{Lift}_v$ for each vertex $v \in V$. For a vertex $v \in V$ the operator $\mathbf{Lift}_v : \Pi \to \Pi$ is defined by

$$\begin{array}{lcll} \mathbf{Lift}_v(\phi)(v) & = & \max\{\min(\{\delta(\phi(v'), \ell) : (v, \ell, v') \in E\}),\ \phi(v)\} & \text{for } v \in \mathrm{V}_{\mathrm{Eve}} \\ \mathbf{Lift}_v(\phi)(v) & = & \max\{\max(\{\delta(\phi(v'), \ell) : (v, \ell, v') \in E\}),\ \phi(v)\} & \text{for } v \in \mathrm{V}_{\mathrm{Adam}} \\ \mathbf{Lift}_v(\phi)(v') & = & \phi(v') & \text{for } v' \neq v \end{array}$$

We reformulate the definition of ϕ being a progress measure using the lift operators as follows.

▶ **Lemma 11.** *The function $\phi : V \to V_{\mathcal{U}_\top}$ is a progress measure if and only if for all $v \in V$ we have $\phi \geq \boldsymbol{Lift}_v(\phi)$.*

The algorithm is given in Algorithm 1, it is an instance of Kleene's fixed point algorithm in a finite complete lattice with a set of inflationary and monotone operators. The next lemma states the properties implying the correctness of this algorithm.

Algorithm 1 The value iteration algorithm.

Data: A game $\mathcal{G}$ of size n with objective Ω and an (n, Ω)-universal linear graph.
for $v \in V$ **do**
  $\phi(v) \leftarrow s_{\text{init}}$;
repeat
  Choose $v \in V$ such that $\phi \not\geq \mathbf{Lift}_v(\phi)$;
  $\phi \leftarrow \mathbf{Lift}_v(\phi)$;
until $\forall v \in V,\ \phi \geq \boldsymbol{Lift}_v(\phi)$;
return ϕ

▶ **Lemma 12.** *The set Π with the pointwise order induced from $V_{\mathcal{U}_\top}$ is a finite complete lattice. For all $v \in V$, the operator $\boldsymbol{Lift}_v$ is inflationary, meaning $\phi \leq \boldsymbol{Lift}_v(\phi)$, and monotone, meaning $\phi \leq \phi'$ implies $\boldsymbol{Lift}_v(\phi) \leq \boldsymbol{Lift}_v(\phi')$.*

▶ **Theorem 13.** *Let Ω be a prefix independent positionally determined objective, $\mathcal{G}$ a game with objective Ω, and $\mathcal{U}$ an (n, Ω)-universal linear graph. The value iteration algorithm outputs the minimal progress measure in time $O(m \cdot n_{\mathcal{U}})$ and space $O(n \cdot \log(n_{\mathcal{U}}))$.*

3 Universal graphs for mean payoff games

We now focus on universal graphs for mean payoff objectives. Let $W \subseteq \mathbb{Z}$ be the set of colours, then

$$\texttt{MeanPayoff}_W = \left\{ w = (w_0, w_1, \dots) : \liminf_\ell \frac{1}{\ell} \sum_{i=0}^{\ell-1} w_i \geq 0 \right\}.$$

The set W is called the set of weights. Mean payoff objectives are prefix independent and positionally determined [11, 32] and extend parity objectives in the following sense. Let d be a natural number, we consider the set of colours $[1, d]$ called priorities. Let us define

$$\text{Parity}_d = \{w : \text{ the largest priority appearing infinitely many times in } w \text{ is even}\}.$$

Let n be a natural number and $W = \{(-n)^p : p \in [1, d]\}$, then for all arenas $\mathcal{A}$ of size at most n and initial vertex v_{init}, Eve has a winning strategy from v_{init} in the game induced by $\mathcal{A}$ and the objective $\texttt{MeanPayoff}_W$ if and only if she has a winning strategy from v_{init} in the game induced by $\mathcal{A}$ and the objective Parity_d.

Let us state in the following theorem the existing results about universal graphs for parity objectives.

▶ **Theorem 14** ([8, 5]). *For all n, d:*

- *There exists an $(n, Parity_d)$-universal graph of size $n^{O(\log(d))}$.*
- *All $(n, Parity_d)$-universal graphs have size at least $n^{\Omega(\log(d))}$.*

The remainder of this paper extends this study to mean payoff objectives, hence for any set of weights W. We consider two parameters on W: the largest weight N in absolute value in Section 4, in other words the case where $W = (-N, N)$, and the number of weights, *i.e.* the cardinality of W, in Section 5.

The construction of a universal graph

Recall that a graph satisfies mean payoff if all maximal paths are infinite and satisfy mean payoff. This can be easily characterised using cycles: a cycle is a path of length ℓ such that $v_0 = v_\ell$. A cycle $(v_0, w_0, v_1) \cdots (v_{\ell-1}, w_{\ell-1}, v_0)$ is negative if its total weight is negative, meaning

$$\sum_{i=0}^{\ell-1} w_i < 0.$$

▶ **Fact 2.** *A graph satisfies mean payoff if and only if it does not contain any sinks or negative cycles.*

Let W be a set of weights. We speak of a W-graph if all the weights in the graph belong to W, and of an (n, W)-graph if additionally its size is at most n. We define a class of graphs called *integer graphs*. An integer W-graph is given by a finite subset A of the integers: the set of vertices is A and for any $v, v' \in A$ and $w \in W$ there is an edge from v to v' labelled by w if $v' - v \leq w$. When defining integer graphs we drop W when it is clear from the context or irrelevant.

Some remarks are in order. First, integer graphs satisfy mean payoff, which follows from the fact that they do not contain sinks (thanks to the self loop labelled by 0) nor negative cycles. The translation $A + p$ of an integer graph A by $p \in \mathbb{Z}$ is isomorphic to A. It also

holds that for A, B two integer W-graphs, A maps into B if and only if there exists $p \in \mathbb{Z}$ such that $A + p \subseteq B$. Finally, integer graphs are linear as defined in the previous section for the natural order on integers.

The key yet simple observation for understanding universal graphs for mean payoff is that given a graph with no negative cycles we can define a distance between vertices: the distance $\mathrm{dist}(v, v')$ from a vertex v to another vertex v' is the smallest sum of the weights along a path from v to v' (when such a path exists).

We let $\Sigma_{W,n}$ denote the set of sums of at most $n-1$ terms from W.

▶ **Lemma 15.** *Let G be a (n, W)-graph satisfying $\mathtt{MeanPayoff}_W$. There exists an integer graph $\mathcal{L}$ of size at most n such that G homomorphically maps into $\mathcal{L}$. Further, $\mathcal{L}$ maps into the integer W-graph $[0, nN)$ and into the integer W-graph $\Sigma_{W,n}$, and two consecutive integers in $\mathcal{L}$ are no more than N apart.*

This lemma has two important consequences. The first is upper bounds on the size of universal graphs for mean payoff objectives for both parameters.

▶ **Corollary 16.**

- *The integer $(-N, N)$-graph $[0, nN)$ is $(n, \mathtt{MeanPayoff}_{(-N,N)})$-universal and has size nN.*
- *For every W of cardinality k, the integer W-graph $\Sigma_{W,n}$ is $(n, \mathtt{MeanPayoff}_W)$-universal and has size $O(n^k)$.*

The second corollary will be useful for proving lower bounds as we will restrict our attention to integer graphs.

▶ **Corollary 17.** *For every $(n, \mathtt{MeanPayoff}_W)$-universal graph, there exists an $(n, \mathtt{MeanPayoff}_W)$-universal integer graph of at most the same size.*

We state here a simple result that we will use several times later on about homomorphisms into integer graphs.

▶ **Lemma 18.** *Let G be a graph, A an integer graph, $\phi : G \to A$ a homomorphism. Consider a cycle $(v_0, w_0, v_1) \cdots (v_{\ell-1}, w_{\ell-1}, v_0)$ in G of total weight 0. Then for $i \in [0, \ell)$, we have $\phi(v_{i+1}) - \phi(v_i) = w_i$, where by convention $v_\ell = v_0$.*

The value iteration algorithm

Let W be a set of weights with $W \subseteq (-N, N)$ and $|W| \leq k$. We can now instantiate Theorem 13 for mean payoff objectives using the two universal graphs constructed in Corollary 16. The first one (parameterised by N) yields exactly the algorithm constructed by Brim, Chaloupka, Doyen, Gentilini, and Raskin [3]: identical data structures and complexity analysis. We note that the two tasks whose complexity were assumed to be constant, namely computing $\delta(s, \ell)$ for s and ℓ, and checking whether $s \leq s'$, are indeed unitary operations as they manipulate numbers of order nN. Assuming these operations take constant time and space (as was done in [3]), we obtain an algorithm with time complexity $O(nmN)$ and space complexity $O(n \log(N))$. The second one (parameterised by k) yields the same algorithm but the time complexity becomes $O(mn^k)$ and the space complexity $O(nk \log(n))$.

▶ **Corollary 19.**

- *There exists an algorithm for solving mean payoff games with weights in $(-N, N)$ of time complexity $O(nmN)$ and space complexity $O(n \log(N))$.*
- *There exists an algorithm for solving mean payoff games with k weights of time complexity $O(mn^k)$ and space complexity $O(nk \log(n))$.*

4 Parametrised by the largest weight

In this section we focus on the largest weight of W in absolute value as parameter, so we fix $W = (-N, N)$. We already explained how to construct an $(n, \texttt{MeanPayoff}_{(-N,N)})$-universal graph of size nN, yielding an algorithm matching the best known complexity. We improve on this upper bound when N is exponential in n, and prove a matching lower bound.

▶ **Theorem 20.**

- *There exists an* $(n, \texttt{MeanPayoff}_{(-N,N)})$*-universal graph of size at most* $2n^2N^{1-1/n}$.
- *All* $(n, \texttt{MeanPayoff}_{(-N,N)})$*-universal graphs have size at least* $N^{1-1/n}$.

Since n is polynomial in the size of the input, one may say that n is "small", while N is exponential in the size of the input when weights are given in binary, hence "large". With this intuition in mind, the multiplicative gap between upper and lower bound is bounded by $2n^2$, hence small.

Let us now discuss the significance of the difference between N to $N^{1-1/n}$. For $N \leq 2^n$ we have $N^{1-1/n} \geq \frac{1}{2}N$ so $N^{1-1/n}$ is essentially linear in N. However when $N \geq 2^{\Omega(n^{1+\varepsilon})}$ for $\varepsilon > 0$ then $2n^2N^{1-1/n} = o(nN)$, so the new universal graph is indeed asymptotically smaller than the previous one. It follows that the new universal graph yields an improved algorithm in this regime. To appreciate the relevance of this condition, let us recall an old result of Frank and Tardos [15] which implies that one can in polynomial time transform a mean payoff game into an equivalent one where $N \leq 2^{4n^3}m^{m+3}$. Our new algorithm improves over the previous one for the range $N \in [2^{\Omega(n^{1+\varepsilon})}, 2^{4n^3}m^{m+3}]$.

▶ **Corollary 21.** *There exists an algorithm for solving mean payoff games with time complexity* $O(mn(nN)^{1-1/n})$ *and with space complexity* $O(n\log(N))$.

This new algorithm improves over the first algorithm when $N = 2^{\Omega(n^{1+\varepsilon})}$, but under this assumption the second algorithm is faster. The improvement is on the space complexity, where our algorithm performs as well as the first algorithm and much better than the second algorithm.

Upper bound

▶ **Theorem 22.** *There exists an* $(n, \texttt{MeanPayoff}_{(-N,N)})$*-universal graph of size*

$$2\left(nN - ((nN)^{1/n} - 1)^n\right).$$

As discussed above, the size of this new universal graph is not always smaller than the first universal graph of size nN, but it is asymptotically smaller when $N = 2^{\Omega(n^2)}$. We now give some intuition for the construction. The construction in Lemma 15 shows that the integer graph $[0, nN)$ is $(n, \texttt{MeanPayoff}_{(-N,N)})$-universal. In this construction the initial vertex v_{init} is always mapped to 0 by the homomorphism. By allowing ourselves to map v_{init} anywhere in the integer graph, we get some slack which enables us to remove some values from $[0, nN)$ while remaining universal. As a drawback we need to double the range to $[0, 2nN)$, which is why this new construction is not always smaller than the original one of size nN.

Lower bound

▶ **Theorem 23.** *Any* $(n, \texttt{MeanPayoff}_{(-N,N)})$*-universal graph has size at least* $N^{1-1/n}$.

5 Parametrised by the number of weights

In this section we focus on the cardinality of W as a parameter. Recall that the very simple integer graph $\Sigma_{W,n}$ is universal and has size n^k. We now give an almost matching lower bound.

▶ **Theorem 24.** *For all k, for n large enough, there exists $W \subseteq \mathbb{Z}$ of cardinality k such that all $(n, \texttt{MeanPayoff}_W)$-universal graphs have size at least $\Omega(n^{k-2})$.*

Theorem 24 follows from the following Lemma. We let $T = 1 + n + n^2 + \cdots + n^{k-2}$ and $W = \left\{1, n, n^2, \ldots, n^{k-2}, -\frac{n-1}{k-1}T\right\}$. Note that W has indeed cardinality k.

▶ **Lemma 25.** *Let $\mathcal{U}$ be an $(n, \texttt{MeanPayoff}_W)$-universal graph. Then $|\mathcal{U}| \geq \left(\frac{n-1}{(k-1)^2}\right)^{\frac{(k-1)^2}{k}}$.*

Conclusions

In this paper we have shown how to extend to mean payoff games the ideas developed for constructing quasipolynomial algorithms for parity games using the combinatorial notion of universal graphs. This yields a family of algorithms for mean payoff games which includes the value iteration algorithm of Brim, Chaloupka, Doyen, Gentilini, and Raskin [3]. We give almost matching lower bounds against two parameters: the largest weight in absolute value and the number of weights. Against the first parameter we obtain an improvement in the case where the largest weight is exponential in the size of the game. The lower bound against the second parameter implies that algorithms based on universal graphs cannot solve mean payoff games in quasipolynomial time.

Our lower bounds show that for pathological sets of weights universal graphs are very large. A more positive note is to consider $W = \{(-n)^p : p \in [1, d]\}$, the set of weights corresponding to parity games: in this case we know that there exist (n, W)-universal graphs of quasipolynomial size (specifically $n^{O(\log(d))}$). This motivates a deeper understanding of the size of $(n, \texttt{MeanPayoff}_W)$-universal graphs: for which sets of weights W do there exist small universal graphs? Is there a meaningful hierarchy between parity and mean payoff games?

This paper represents a new milestone in the fruitful line of research constructing algorithms for solving games using universal graphs. We have construct a family of value iteration algorithms with very efficient space complexity which can be instantiated for any positionally determined objective. So far only parity and mean payoff objectives have been studied under this light. There are many more important positionally determined objectives, such as the Rabin objectives which play an important role in LTL synthesis, and the combination of mean payoff and parity objectives [9]. Another direction is to construct algorithms for games played on subclasses of graphs such as planar graphs, bounded clique or tree width graphs.

References

1 Xavier Allamigeon, Pascal Benchimol, Stéphane Gaubert, and Michael Joswig. Combinatorial simplex algorithms can solve mean payoff games. *SIAM Journal on Optimization*, 24(4):2096–2117, 2014. doi:10.1137/140953800.

2 Mikołaj Bojańczyk and Wojciech Czerwiński. An automata toolbox, February 2018. URL: https://www.mimuw.edu.pl/~bojan/papers/toolbox-reduced-feb6.pdf.

3 Lubos Brim, Jakub Chaloupka, Laurent Doyen, Raffaella Gentilini, and Jean-François Raskin. Faster algorithms for mean-payoff games. *Formal Methods in System Design*, 38(2):97–118, 2011. doi:10.1007/s10703-010-0105-x.

4 Cristian S. Calude, Sanjay Jain, Bakhadyr Khoussainov, Wei Li, and Frank Stephan. Deciding parity games in quasipolynomial time. In *STOC*, pages 252–263, 2017. doi:10.1145/3055399.3055409.

5 Thomas Colcombet and Nathanaël Fijalkow. Parity games and universal graphs. *CoRR*, abs/1810.05106, 2018. arXiv:1810.05106.

6 Thomas Colcombet and Nathanaël Fijalkow. Universal graphs and good for games automata: New tools for infinite duration games. In *FoSSaCS*, pages 1–26, 2019. doi:10.1007/978-3-030-17127-8_1.

7 Carlo Comin and Romeo Rizzi. Improved pseudo-polynomial bound for the value problem and optimal strategy synthesis in mean payoff games. *Algorithmica*, 77(4):995–1021, 2017. doi:10.1007/s00453-016-0123-1.

8 Wojciech Czerwiński, Laure Daviaud, Nathanaël Fijalkow, Marcin Jurdziński, Ranko Lazić, and Paweł Parys. Universal trees grow inside separating automata: Quasi-polynomial lower bounds for parity games. In *SODA*, pages 2333–2349, 2019. doi:10.1137/1.9781611975482.142.

9 Laure Daviaud, Marcin Jurdziński, and Ranko Lazić. A pseudo-quasi-polynomial algorithm for mean-payoff parity games. In *LICS*, pages 325–334, 2018. doi:10.1145/3209108.3209162.

10 Dani Dorfman, Haim Kaplan, and Uri Zwick. A faster deterministic exponential time algorithm for energy games and mean payoff games. In *ICALP*, pages 114:1–114:14, 2019. doi:10.4230/LIPIcs.ICALP.2019.114.

11 Andrzej Ehrenfeucht and Jan Mycielski. Positional strategies for mean payoff games. *International Journal of Game Theory*, 109(8):109–113, 1979. doi:10.1007/BF01768705.

12 John Fearnley, Sanjay Jain, Sven Schewe, Frank Stephan, and Dominik Wojtczak. An ordered approach to solving parity games in quasi polynomial time and quasi linear space. In *SPIN*, pages 112–121, 2017.

13 Nathanaël Fijalkow. An optimal value iteration algorithm for parity games. *CoRR*, abs/1801.09618, 2018. arXiv:1801.09618.

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

15 András Frank and Éva Tardos. An application of simultaneous diophantine approximation in combinatorial optimization. *Combinatorica*, 7(1):49–65, 1987. doi:10.1007/BF02579200.

16 Ofer Freedman, Paweł Gawrychowski, Patrick K. Nicholson, and Oren Weimann. Optimal distance labeling schemes for trees. In *PODC*, pages 185–194. ACM, 2017.

17 Paweł Gawrychowski, Fabian Kuhn, Jakub Łopuszański, Konstantinos Panagiotou, and Pascal Su. Labeling schemes for nearest common ancestors through minor-universal trees. In *SODA*, pages 2604–2619. SIAM, 2018.

18 Vladimir A. Gurvich, Aleksander V. Karzanov, and Leonid G. Khachiyan. Cyclic games and an algorithm to find minimax cycle means in directed graphs. *USSR Computational Mathematics and Mathematical Physics*, 28:85–91, 1988.

19 Marcin Jurdziński and Ranko Lazić. Succinct progress measures for solving parity games. In *LICS*, pages 1–9, 2017. doi:10.1109/LICS.2017.8005092.

20 Marcin Jurdziński and Rémi Morvan. A universal attractor decomposition algorithm for parity games. *CoRR*, abs/2001.04333, 2020. arXiv:2001.04333.

21 Gil Kalai. A subexponential randomized simplex algorithm (extended abstract). In *STOC*, pages 475–482, 1992. doi:10.1145/129712.129759.

22 Gil Kalai. Linear programming, the simplex algorithm and simple polytopes. *Math. Program.*, 79:217–233, 1997. doi:10.1007/BF02614318.

23 Sampath Kannan, Moni Naor, and Steven Rudich. Implicit representation of graphs. *SIAM Journal on Discrete Mathematics*, 5(4):596–603, 1992.

24 Karoliina Lehtinen. A modal-μ perspective on solving parity games in quasi-polynomial time. In *LICS*, pages 639–648, 2018.

25 Karoliina Lehtinen, Sven Schewe, and Dominik Wojtczak. Improving the complexity of parys' recursive algorithm. *CoRR*, abs/1904.11810, 2019. arXiv:1904.11810.

26 Y.M. Lifshits and D.S. Pavlov. Potential theory for mean payoff games. *Journal of Mathematical Sciences*, 145:4967–4974, 2007. doi:10.1007/s10958-007-0331-y.

27 Jivr'i Matouvsek, Micha Sharir, and Emo Welzl. A subexponential bound for linear programming. *Algorithmica*, 16(4/5):498–516, 1996. doi:10.1007/BF01940877.

28 Paweł Parys. Parity games: Zielonka's algorithm in quasi-polynomial time. In *MFCS*, pages 10:1–10:13, 2019. doi:10.4230/LIPIcs.MFCS.2019.10.

29 Paweł Parys. Parity games: Another view on lehtinen's algorithm. In *CSL*, pages 32:1–32:15, 2020. doi:10.4230/LIPIcs.CSL.2020.32.

30 Steve Smale. Mathematical problems for the next century. *The Mathematical Intelligencer*, 20(2):7–15, 1998. doi:10.1007/BF03025291.

31 Wiesław Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theoretical Computer Science*, 200(1-2):135–183, 1998. doi:10.1016/S0304-3975(98)00009-7.

32 Uri Zwick and Mike Paterson. The complexity of mean payoff games on graphs. *Theoretical Computer Science*, 158(1&2):343–359, 1996. doi:10.1016/0304-3975(95)00188-3.