New Algorithms for Combinations of Objectives using Separating Automata

Ashwani Anand
Chennai Mathematical Institute, Chennai, India

Nathanaël Fijalkow
CNRS, LaBRI, Bordeaux, France
The Alan Turing Institute, London, United Kingdom

Aliénor Goubault-Larrecq
ENS Lyon, Lyon, France

Jérôme Leroux
CNRS, LaBRI, Bordeaux, France

Pierre Ohlmann
Université de Paris, Paris, France

The notion of separating automata was introduced by Bojańczyk and Czerwiński for understanding the first quasipolynomial time algorithm for parity games. In this paper we show that separating automata is a powerful tool for constructing algorithms solving games with combinations of objectives. We construct two new algorithms: the first for disjunctions of parity and mean payoff objectives, matching the best known complexity, and the second for disjunctions of mean payoff objectives, improving on the state of the art. In both cases the algorithms are obtained through the construction of small separating automata, using as black boxes the existing constructions for parity objectives and for mean payoff objectives.

1 Introduction

The notion of separating automata was introduced by Bojańczyk and Czerwiński [1] to give a streamlined presentation of the first quasipolynomial time algorithm for parity games due to Calude, Jain, Khoussainov, Li, and Stephan [2]. The first observation made by Bojańczyk and Czerwiński was that the statistics used in that algorithm can be computed by a finite deterministic safety automaton reading a path in the game. They showed that the property making this particular automaton useful for solving parity games is that, roughly speaking, it separates positional winning paths from losing paths: they called this property separating. The second observation of Bojańczyk and Czerwiński was that separating automata yield a natural reduction to safety games, in other words the construction of a separating automaton induces an algorithm whose complexity depends on the size of the automaton.

The notion of separating automata has been further studied in the context of parity games: Czerwiński, Daviaud, Fijalkow, Jurdziński, Lazić, and Parys [7] showed that two other quasipolynomial time algorithms can also be presented using the construction of a separating automaton. The main technical result of [7] is that separating automata are in some sense equivalent to the notion of universal trees at the heart of the second quasipolynomial time algorithm by Jurdziński and Lazić [12] and formalised by Fijalkow [9]. The consequence of this equivalence is a quasipolynomial lower bound on the size of separating automata for parity objectives.

Going beyond parity games, Colcombet and Fijalkow [5, 6] introduced universal graphs and showed an equivalence result between separating automata and universal graphs for any positionally determined objective. This paves the way for using separating automata for other classes of objectives. The first work in that direction is due to Fijalkow, Ohlmann, and Gawrychowski [10], who obtained matching upper and lower bounds on the size of separating automata for mean payoff objectives, matching the best known (deterministic) complexity for solving mean payoff games.

D. Bresolin and P. Ganty (Eds.): 12th International Symposium
on Games, Automata, Logics, and Formal Verification (GandALF 2021)
EPTCS 346, 2021, pp. 227–240, doi:10.4204/EPTCS.346.15

The goal of this paper is to show how to construct separating automata for combinations of objectives, thereby obtaining new algorithms for solving the corresponding games. We will consider two subclasses combining parity and mean payoff objectives, with the following goal: rather than constructing separating automata from scratch, we want to define constructions using separating automata for the atomic objectives as black boxes. In other words, we assume the existence of separating automata for parity objectives (provided in [7], see also [5]) and for mean payoff objectives (provided in [10]), and construct separating automata for combinations of these classes. An important benefit of this approach is its simplicity: as we will see, both constructions and their correctness proofs are rather short and focus on the interactions between the objectives.

Section 2 introduces separating automata and shows how they yield algorithms by reduction to safety games. The two classes of objectives we consider are disjunctions of parity and mean payoff objectives in Section 3, and disjunctions of mean payoff objectives in Section 4. We refer to the subsections 3.2 and 4.3 for related work on solving these games.

2 Preliminaries

We write $[i,j]$ for the interval $\{i,i+1,\dots,j-1,j\}$, and use parentheses to exclude extremal values, so for instance $[i,j)$ is $\{i,i+1,\dots,j-1\}$. 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). The empty word is ε.

2.1 Graphs

Graphs We consider edge labelled directed graphs: a graph G is given by a (finite) set V of vertices and a (finite) set $E \subseteq V \times C \times V$ of edges, with C a set of colours, so we write $G = (V,E)$. An edge (v,c,v') is from the vertex v to the vertex v' and is labelled by the colour c. We sometimes refer to $V(G)$ for V and $E(G)$ for E to avoid any ambiguity. The size of a graph is its number of vertices. A vertex v for which there exists no outgoing edges $(v,c,v') \in E$ is called a sink.

Paths A path π is a (finite or infinite) sequence

$$\pi = v_0c_0v_1c_1v_2\dots$$

where for all i we have $(v_i,c_i,v_{i+1}) \in E$. If it is finite, that is, $\pi = v_0c_0\dots c_{i-1}v_i$, we call $\text{len}(\pi) = i \geq 0$ the length of π, and use $\text{last}(\pi)$ to denote the last vertex v_i. The length of an infinite path is $\text{len}(\pi) = \infty$. We say that π starts from v_0 or is a path from v_0, and in the case where π is finite we say that π is a path ending in $\text{last}(\pi)$ or simply a path to $\text{last}(\pi)$. We say that v' is reachable from v if there exists a path from v to v'. We let $\pi_{\leq i}$ denote the prefix of π of length i, meaning $\pi_{\leq i} = v_0c_0v_1\dots c_{i-1}v_i$. A cycle is a path from a vertex to itself of length at least one.

We use $\text{Path}_{\text{fin}}(G), \text{Path}_\infty(G)$ and $\text{Path}(G)$ to denote respectively the sets of finite paths of G, infinite paths of G, and their union. We sometimes drop G when it is clear from context. For $v_0 \in V$ we also use $\text{Path}_{\text{fin}}(G,v_0), \text{Path}_\infty(G,v_0)$ and $\text{Path}(G,v_0)$ to refer to the sets of paths starting from v_0. We use $\text{col}(\pi)$ to denote the (finite or infinite) sequence of colours $c_0c_1\dots$ induced by π.

Objectives An objective is a set $\Omega \subseteq C^\omega$ of infinite sequences of colours. We say that a sequence of colours belonging to Ω satisfies Ω, and extend this terminology to infinite paths: π satisfies Ω if $\mathrm{col}(\pi) \in \Omega$.

Definition 1 (Graphs satisfying an objective)**.** *Let Ω be an objective and G a graph. We say that G satisfies Ω if all infinite paths in G satisfy Ω.*

Safety automata A deterministic safety automaton over the alphabet C is given by a finite set of states Q, an initial state $q_0 \in Q$, and a transition function $\delta : Q \times C \to Q$, so we write $\mathscr{A} = (Q, q_0, \delta)$. Note that δ is a partial function, meaning that it may be that $\delta(q,c)$ is undefined for some $q,c \in Q \times C$. Such an automaton induces a graph whose set of vertices is Q and set of edges is $E = \{(q, \delta(q,c)) : q \in Q, c \in C\}$; we therefore use the terminology for graphs to speak about automata, and identify an automaton and the graph it induces. Since we are only considering deterministic safety automata in this paper, we omit the adjectives deterministic and safety and simply speak of an automaton. We extend δ to sequences of colours by the formulas $\delta^*(q,\varepsilon) = q$ and $\delta^*(q,wc) = \delta(\delta^*(q,w),c)$. The language recognised by an automaton $\mathscr{A}$ is $L(\mathscr{A})$ defined by

$$L(\mathscr{A}) = \{\mathrm{col}(\pi) : \pi \text{ infinite path from } q_0\}.$$

Note that if w is a finite prefix of a word in $L(\mathscr{A})$, then $\delta^*(q_0,w)$ is well defined.

2.2 Games

Arenas An arena is given by a graph G together with a partition $V = \mathrm{V}_{\mathrm{Eve}} \uplus \mathrm{V}_{\mathrm{Adam}}$ of its set of vertices describing which player controls each vertex.

Games A game is given by an arena and an objective Ω. We often let $\mathscr{G}$ denote a game, its size is the size of the underlying graph. It is played as follows. A token is initially placed on some vertex v_0, 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 either forever and describing an infinite path or until reaching a sink.

We say that a path is winning[1] if it is infinite and satisfies Ω, or finite and ends in a sink. The definition of a winning path includes the following usual convention: if a player cannot move they lose, in other words sinks controlled by Adam are winning (for Eve) and sinks controlled by Eve are losing.

We extend the notations $\mathrm{Path}_{\mathrm{fin}}, \mathrm{Path}_\infty$ and Path to games by considering the underlying graph.

Strategies A strategy in $\mathscr{G}$ is a partial map $\sigma : \mathrm{Path}_{\mathrm{fin}}(\mathscr{G}) \to E$ such that $\sigma(\pi)$ is an outgoing edge of $\mathrm{last}(\pi)$ when it is defined. We say that a path $\pi = v_0 c_0 v_1 \dots$ is consistent with σ if for all $i < \mathrm{len}(\pi)$, if $v_i \in \mathrm{V}_{\mathrm{Eve}}$ then σ is defined over $\pi_{\leq i}$ and $\sigma(\pi_{\leq i}) = (v_i, c_i, v_{i+1})$. A consistent path with σ is maximal if it is not the strict prefix of a consistent path with σ (in particular infinite consistent paths are maximal).

A strategy σ is winning from v_0 if all maximal paths consistent with σ are winning. Note that in particular, if a finite path π is consistent with a winning strategy σ and ends in a vertex which belongs to Eve, then σ is defined over π. We say that v_0 is a winning vertex of $\mathscr{G}$ or that Eve wins from v in $\mathscr{G}$ if there exists a winning strategy from v_0.

[1]We always take the point of view of Eve, so winning means winning for Eve, and similarly a strategy is a strategy for Eve.

Positional strategies Positional strategies make decisions only considering the current vertex. Such a strategy is given by $\widehat{\sigma} : \mathrm{V}_{\mathrm{Eve}} \to E$. A positional strategy induces a strategy $\sigma : \mathrm{Path}_{\mathrm{fin}} \to E$ from any vertex v_0 by setting $\sigma(\pi) = \widehat{\sigma}(\mathrm{last}(\pi))$ when $\mathrm{last}(\pi) \in \mathrm{V}_{\mathrm{Eve}}$.

Definition 2. *We say that an objective Ω is positionally determined if for every game with objective Ω and vertex v_0, if Eve wins from v_0 then there exists a positional winning strategy from v_0.*

Given a game $\mathcal{G}$, a vertex v_0, and a positional strategy σ we let $\mathcal{G}[\sigma, v_0]$ denote the graph obtained by restricting $\mathcal{G}$ to vertices reachable from v_0 by playing σ and to the moves prescribed by σ. Formally, the set of vertices and edges is

$$\begin{array}{rcl} V[\sigma,v_0] & = & \{v \in V : \text{there exists a path from } v_0 \text{ to } v \text{ consistent with } \sigma\}, \\ E[\sigma,v_0] & = & \{(v,c,v') \in E : v \in \mathrm{V}_{\mathrm{Adam}} \text{ or } (v \in \mathrm{V}_{\mathrm{Eve}} \text{ and } \sigma(v) = (v,c,v'))\} \\ & \cap & V[\sigma,v_0] \times C \times V[\sigma,v_0]. \end{array}$$

In this paper we will consider prefix independent objectives for technical convenience.

Definition 3. *We say that an objective Ω is prefix independent if for all $u \in C^*$ and $v \in C^\omega$, we have $uv \in \Omega$ if and only if $v \in \Omega$.*

Lemma 1. *Let Ω be a prefix independent objective, $\mathcal{G}$ a game, v_0 a vertex, and σ a positional strategy. Then σ is winning from v_0 if and only if the graph $\mathcal{G}[\sigma, v_0]$ satisfies Ω and does not contain any sink controlled by Eve.*

Solving games

Decision problem The decision problem we consider in this paper, called solving a game, is the following: given a game $\mathcal{G}$ and an initial vertex v_0, does Eve have a winning strategy from v_0 in $\mathcal{G}$? Each objective yields a class of games, so we speak for instance of "solving mean payoff games".

Computational model The complexity of solving a game depends on a number of parameters originating either from the underlying graph or the objective. Some of the objectives involve rational numbers. The typical parameters from the underlying graph are the number n of vertices and the number m of edges.

We use the classical Random Access Model (RAM) of computation with fixed word size, which is the size of a memory cell on which arithmetic operations take constant time. We specify for each class of objectives the word size.

Reduction to safety games

Safety games The safety objective `Safe` is defined over the set of colours $C = \{\varepsilon\}$ by $\texttt{Safe} = \{\varepsilon^\omega\}$: in words, all infinite paths are winning, so losing for Eve can only result from reaching a sink that she controls. Since there is a unique colour, when manipulating safety games we ignore the colour component for edges.

Note that in safety games, strategies can equivalently be defined as maps to V (and not E): only the target of an edge matters when the source is fixed, since there is a unique colour. We use this abuse of notations for the sake of simplicity and conciseness.

Theorem 1. *There exists an algorithm in the RAM model with word size $w = \log(n)$ computing the set of winning vertices of a safety game running in time $O(m)$ and space $O(n)$.*

Reduction using safety automata Let $\mathscr{A} = (Q, q_0, \delta)$ be an automaton and $\mathscr{G}$ a game with objective Ω. We define the chained game $\mathscr{G} \triangleright \mathscr{A}$ as the safety game with vertices $V' = \mathrm{V}'_{\mathrm{Eve}} \uplus \mathrm{V}'_{\mathrm{Adam}}$ and edges E' given by

$$\begin{array}{lcl} \mathrm{V}'_{\mathrm{Eve}} & = & (\mathrm{V}_{\mathrm{Eve}} \times Q) \cup \{\bot\}, \\ \mathrm{V}'_{\mathrm{Adam}} & = & \mathrm{V}_{\mathrm{Adam}} \times Q, \\ E' & = & \{((v,q), \varepsilon, (v', \delta(q,c))) : q \in Q, (v,c,v') \in E, \delta(q,c) \text{ is defined}\} \\ & \cup & \{((v,q), \varepsilon, \bot) : q \in Q, (v,c,v') \in E, \delta(q,c) \text{ is not defined}\}. \end{array}$$

In words, from $(v,q) \in V \times Q$, the player whom v belongs to chooses an edge $(v,c,v') \in E$, and the game progresses to $(v', \delta(q,c))$ if q has an outgoing edge with colour c in $\mathscr{A}$, and to $\bot$ otherwise, which is losing for Eve.

Note that the obtained game $\mathscr{G} \triangleright \mathscr{A}$ is a safety game: Eve wins if she can play forever or end up in a sink controlled by Adam.

Lemma 2. *Let Ω be an objective, $\mathscr{G}$ a game with objective Ω, $\mathscr{A}$ an automaton, and $v_0 \in V$. If $\mathscr{A}$ satisfies Ω and Eve wins from (v_0, q_0) in $\mathscr{G} \triangleright \mathscr{A}$, then Eve wins from v_0 in $\mathscr{G}$.*

Proof. Let σ' be a winning strategy from (v_0, q_0) in $\mathscr{G} \triangleright \mathscr{A}$. We construct a strategy σ from v_0 in $\mathscr{G}$ which simulates σ' in the following sense, which we call the simulation property:

$$\begin{gathered} \text{for any path } \pi = v_0 c_0 \dots c_{i-1} v_i \text{ in } \mathscr{G} \text{ consistent with } \sigma, \\ \text{there exists a path } \pi' = (v_0, q_0) \dots (v_i, q_i) \text{ in } \mathscr{G} \triangleright \mathscr{A} \text{ consistent with } \sigma'. \end{gathered}$$

We define σ over finite paths $v_0 c_0 \dots c_{i-1} v_i$ with $v_i \in \mathrm{V}_{\mathrm{Eve}}$ by induction over i, so that the simulation property holds. For $i = 0$, $\pi' = (v_0, q_0)$ is a path in $\mathscr{G} \triangleright \mathscr{A}$ consistent with σ'. Let $\pi = v_0 c_0 \dots c_{i-1} v_i$ be a path in $\mathscr{G}$ consistent with σ and $v_i \in \mathrm{V}_{\mathrm{Eve}}$, we want to define $\sigma(\pi)$. Thanks to the simulation property there exists a path $\pi' = (v_0, q_0) \dots (v_i, q_i)$ in $\mathscr{G} \triangleright \mathscr{A}$ consistent with σ'. Then $(v_i, q_i) \in \mathrm{V}'_{\mathrm{Eve}}$, and since σ' is winning it is defined over π', let us write $\sigma'(\pi') = (v_{i+1}, \delta(q_i, c_i))$ with $(v_i, c_i, v_{i+1}) \in E$. We set $\sigma(\pi) = v_{i+1}$.

To conclude the definition of σ we need to show that the simulation property extends to paths of length $i+1$. Let $\pi_{i+1} = \pi\ c_i v_{i+1} = v_0 c_0 \dots v_i c_i v_{i+1}$ be consistent with σ. We apply the simulation property to π to construct $\pi' = (v_0, q_0) \dots (v_i, q_i)$ a path in $\mathscr{G} \triangleright \mathscr{A}$ consistent with σ'. Let us consider $\pi'_{+1} = \pi'\ (v_{i+1}, \delta(q_i, c_i))$ with $(v_i, c_i, v_{i+1}) \in E$, we claim that π'_{+1} is consistent with σ'. Indeed, if $v_i \in \mathrm{V}_{\mathrm{Adam}}$, there is nothing to prove, and if $v_i \in \mathrm{V}_{\mathrm{Eve}}$, this holds by construction: since $\sigma(\pi) = v_{i+1}$ we have $\sigma'(\pi') = (v_{i+1}, \delta(q_i, c_i))$. This concludes the inductive proof of the simulation property together with the definition of σ.

We now prove that σ is a winning strategy from v_0. Let $\pi = v_0 c_0 v_1 \dots$ be a maximal consistent path with σ, and let $\pi' = (v_0, q_0)(v_1, q_1) \dots$ be the corresponding path in $\mathscr{G} \triangleright \mathscr{A}$ consistent with σ'. Let us first assume that π is finite, and let $v_i = \mathrm{last}(\pi)$. If $v_i \in \mathrm{V}_{\mathrm{Eve}}$, then by construction σ is defined over π, so the path $v_0 c_0 \dots v_i c_i v_{i+1}$, where $\sigma(\pi) = v_{i+1}$ is consistent with σ, contradicting maximality of π. If however $v_i \in \mathrm{V}_{\mathrm{Adam}}$, then by maximality of σ, v_i must be a sink, hence π is winning. Now if π is infinite then so is π', and then $q_0 c_0 q_1 c_1 \dots$ is a path in $\mathscr{A}$, so $\mathrm{col}(\pi) = c_0 c_1 \cdots \in L(\mathscr{A}) \subseteq \Omega$, and π is winning. We conclude that σ is a winning strategy from v_0 in $\mathscr{G}$. □

It is not hard to see that if $\mathscr{A}$ is deterministic and recognises exactly Ω, then $\mathscr{G}$ and $\mathscr{G} \triangleright \mathscr{A}$ are equivalent. However, for many objectives Ω (for instance, parity or mean payoff objectives), a simple topological argument shows that such deterministic automata do not exist. Separating automata are defined by introducing n as a parameter, and relaxing the condition $L(\mathscr{A}) = \Omega$ to the weaker condition $\Omega^{|n} \subseteq L(\mathscr{A}) \subseteq \Omega$, where $\Omega^{|n}$ is the set of infinite sequence of colours that label paths from graphs of size at most n satisfying Ω. Formally[2],

$$\Omega^{|n} = \{\mathrm{col}(\pi) \mid \pi \in \mathrm{Path}_\infty(G), G \text{ has size at most } n \text{ and satisfies } \Omega\}.$$

Definition 4. *An (n,Ω)-separating automaton $\mathscr{A}$ is an automaton such that*

$$\Omega^{|n} \subseteq L(\mathscr{A}) \subseteq \Omega.$$

The definition given here differs from the original one given by Bojańczyk and Czerwiński [1], who use a different relaxation $\Omega_{|n}$ satisfying $\Omega^{|n} \subseteq \Omega_{|n} \subseteq \Omega$. Therefore a separating automaton in the sense of [1] is also a separating automaton in our sense (but not conversely).

Theorem 2. *Let Ω be a positionally determined objective, $\mathscr{A}$ an (n,Ω)-separating automaton, $\mathscr{G}$ a game of size n, and $v_0 \in V$. Then Eve wins from v_0 in $\mathscr{G}$ if and only if she wins from (v_0,q_0) in $\mathscr{G} \triangleright \mathscr{A}$.*

Proof. The "if" directly follows from Lemma 2. Conversely, assume that Eve wins from v_0 in $\mathscr{G}$, and let σ be a strategy from v_0 in $\mathscr{G}$, which we choose to be positional. As explained in the definition of safety games, without loss of generality we can see σ as a function $\sigma : \mathrm{V}_{\mathrm{Eve}} \to V$. We define a positional strategy $\sigma' : \mathrm{V}'_{\mathrm{Eve}} \to V'$ by

$$\sigma'(v,q) = (\sigma(v), \delta(q,c)) \text{ with } (v,c,v') \in E.$$

Let $\pi' = (v_0,q_0)\dots(v_i,q_i)$, with for all $j \le i, e_j = (v_j,c_j,v_{j+1}) \in E$, be a finite path in $\mathscr{G} \triangleright \mathscr{A}$ consistent with σ', and let $\pi = v_0c_0\dots c_iv_{i+1}$. Then by definition of σ', π is consistent with σ, which rephrases a being a path in $\mathscr{G}[\sigma,v_0]$. Now, $\mathscr{G}[\sigma,v_0]$ is a graph satisfying Ω, so $\mathrm{col}(\pi) = c_0\dots c_i$ is a prefix of a word in $\Omega^{|n} \subseteq L(\mathscr{A})$. This implies that $\delta^*(q_0,c_0\dots c_i)$ is well defined, hence so is $\delta(q_i,c_i)$. In particular, σ' induces a strategy in $\mathscr{G} \triangleright \mathscr{A}$ from (v_0,q_0). We now prove that it is winning.

Let $\pi' = (v_0,q_0)\dots$ be a maximal path in $\mathscr{G}$ consistent with σ', which we assume to be finite for contradiction. Let $\mathrm{last}(\pi') = (v_i,q_i)$ be a sink, and define $\pi = v_0c_0\dots c_{i-1}v_i$. By definition of π', it holds that π is a path in $\mathscr{G}$ which is consistent with σ. Since π is finite and σ is winning v_i cannot be a sink, it has an outgoing edge $e_i = (v_i,c,v') \in E$. Then $((v_i,q_i),\varepsilon,(v_{i+1},\delta(q_i,c_i)) \in E'$, so (v_i,q_i) is not a sink: a contradiction. Hence π' is an infinite path in the safety game $\mathscr{G} \triangleright \mathscr{A}$; it is winning by definition. ☐

Existing constructions for parity and mean payoff games

Parity games The parity objective is defined over the set of colours $[0,d] \subseteq \mathbb{N}$, as follows:

$$\mathtt{Parity}_d = \{w \in [0,d]^\omega : \text{the largest priority appearing infinitely many times in } w \text{ is even}\}.$$

As mentioned above, the definition of separating automata given in [1] is slightly different, but the result below indeed holds for both definitions, as explained in [7].

Theorem 3 ([1, 7]). *Let $n,d \in \mathbb{N}$. There exists an $(n,\mathtt{Parity}_d)$-separating automaton of size*

$$O\left(n \cdot \binom{\lceil \log(n) \rceil + d/2 - 1}{\lceil \log(n) \rceil}\right).$$

[2]Here we use the fact that Ω is prefix independent to simplify the definition: we are considering infinite paths from any vertex of the graph. To extend this definition beyond prefix independent objectives we would need to fix an initial vertex.

Mean payoff games The mean payoff objective is defined over the set of colours $[-N,N] \subseteq \mathbb{Z}$, as follows:

$$\mathtt{MeanPayoff}_N = \left\{ w \in [-N,N]^\omega : \liminf_n \frac{1}{n} \sum_{i=1}^{n} w_i \geq 0 \right\}.$$

Theorem 4 ([10])**.** *Let $n,N \in \mathbb{N}$. There exists an $(n,\mathtt{MeanPayoff}_N)$-separating automaton of size $O(nN)$.*

3 Disjunction of parity and mean payoff

We define the objective $\mathtt{Parity}_d \vee \mathtt{MeanPayoff}_N$ referred to as 'disjunction of parity and mean payoff'. The set of colours is $[0,d] \times [-N,N]$. For $w \in ([0,d] \times [-N,N])^\omega$ we write $w_P \in [0,d]^\omega$ for the projection on the first component and $w_{MP} \in [-N,N]^\omega$ for the projection on the second component.

$$\mathtt{Parity}_d \vee \mathtt{MeanPayoff}_N = \{w \in ([0,d] \times [-N,N])^\omega : w_P \in \mathtt{Parity}_d \vee w_{MP} \in \mathtt{MeanPayoff}_N\}.$$

We refer to Subsection 3.2 for a discussion on existing results.

Theorem 5 ([3])**.** *Disjunctions of parity and mean payoff objectives are prefix independent and positionally determined.*

3.1 Separating automata for disjunctions of parity and mean payoff objectives

Theorem 6. *Let $n,d,N \in \mathbb{N}$. Let $\mathscr{A}_P$ an $(n,\mathtt{Parity}_d)$-separating automaton, $\mathscr{A}_{MP}$ an $(n,\mathtt{MeanPayoff}_N)$-separating automaton. Then there exists an $(n,\mathtt{Parity}_d \vee \mathtt{MeanPayoff}_N)$-separating automaton of size $O(d \cdot |\mathscr{A}_P| \cdot |\mathscr{A}_{MP}|)$.*

Proof. Let us write $\mathscr{A}_P = (Q_P, q_{0,P}, \delta_P)$ and $\mathscr{A}_{MP} = (Q_{MP}, q_{0,MP}, \delta_{MP})$. We define a deterministic automaton $\mathscr{A}_{P \vee MP}$: the set of states is $[0,d] \times Q_P \times Q_{MP}$, the initial state is $(d, q_{0,P}, q_{0,MP})$, and the transition function is

$$\delta((p,q_P,q_{MP}),(p',w)) = \begin{cases} (\max(p,p'), q_P, \delta_{MP}(q_{MP},w)) & \text{if } \delta_{MP}(q_{MP},w) \text{ is defined,} \\ (0, \delta_P(q_P, \max(p,p')), q_{0,MP}) & \text{if } \delta_{MP}(q_{,MP},w) \text{ is not defined.} \end{cases}$$

Intuitively: $\mathscr{A}_{P \vee MP}$ simulates the automaton $\mathscr{A}_{MP}$, storing the maximal priority seen since the last reset (or from the beginning). If the automaton $\mathscr{A}_{MP}$ rejects, the automaton resets, which means two things: it simulates one transition of the automaton $\mathscr{A}_P$ using the stored priority, and resets the state of $\mathscr{A}_{MP}$ to its initial state. The automaton $\mathscr{A}_{P \vee MP}$ rejects only if $\mathscr{A}_P$ rejects during a reset.

We now prove that $\mathscr{A}_{P \vee MP}$ is an $(n,\mathtt{Parity}_d \vee \mathtt{MeanPayoff}_N)$-separating automaton.

- $L(\mathscr{A}_{P \vee MP}) \subseteq \mathtt{Parity}_d \vee \mathtt{MeanPayoff}_N$. Let π be an infinite path accepted by $\mathscr{A}_{P \vee MP}$, we distinguish two cases by looking at the run of π. We extend the previous notation for projections: for a path π, we write π_P for its projection on the first component and π_{MP} for its projection on the second component.
  - If the run is reset finitely many times, let us write $\pi = \pi'\pi''$ where π' is finite and π'_{MP} rejected by $\mathscr{A}_{MP}$, and π'' is infinite and π''_{MP} is accepted by $\mathscr{A}_{MP}$. Since $\mathscr{A}_{MP}$ satisfies $\mathtt{MeanPayoff}_N$, this implies that π''_{MP} satisfies $\mathtt{MeanPayoff}_N$, so by prefix independence, π_{MP} satisfies $\mathtt{MeanPayoff}_N$ hence π satisfies $\mathtt{Parity}_d \vee \mathtt{MeanPayoff}_N$.

- If the run is reset infinitely many times, we may find an infinite decomposition $\pi = \pi^1\pi^2\dots$ where for each $i \in \mathbb{N}$, the path π^i_{MP} is rejected by $\mathscr{A}_{MP}$, and its proper prefixes are not. Let p_i be the maximum priority appearing in π^i_P, the run of π over $\mathscr{A}_{P\vee MP}$ induces a run of $p_1p_2\dots$ over $\mathscr{A}_P$. Since $\mathscr{A}_P$ satisfies $\mathtt{Parity}_d$, this implies that $p_1p_2\dots$ satisfies $\mathtt{Parity}_d$. By definition of the p_i's, this implies that π_P satisfies $\mathtt{Parity}_d$ so a fortiori π satisfies $\mathtt{Parity}_d \vee \mathtt{MeanPayoff}_N$.

- $(\mathtt{Parity}_d \vee \mathtt{MeanPayoff}_N)^{|n} \subseteq L(\mathscr{A}_{P\vee MP})$. Let G be a graph satisfying $\mathtt{Parity}_d \vee \mathtt{MeanPayoff}_N$ of size at most n, we need to show that all infinite paths of G are accepted by $\mathscr{A}_{P\vee MP}$.

  We construct a graph G_{MP} over the set of colours $[-N,N]$ and a graph G_P over the set of colours $[0,d]$. Both use the same set of vertices as G. We prove that the graph G_{MP} satisfies $\mathtt{MeanPayoff}_N$, which is used to prove that G_P satisfies $\mathtt{Parity}_d$.

  - **The graph** G_{MP}**.** There is an edge $(v,w,v') \in E(G_{MP})$ if there exists $p \in [0,d]$ such that $e = (v,(p,w),v') \in E(G)$, and either p is odd or p is even and e is not contained in any negative cycle with maximum priority p.

    We claim that G_{MP} satisfies $\mathtt{MeanPayoff}_N$. Assume towards contradiction that G_{MP} contains a negative cycle C_{MP}. It induces a negative cycle C in G, by definition of G_{MP} necessarily the maximum priority in C is odd. Hence C is a negative odd cycle in G, a contradiction.
  - **The graph** G_P**.** There is an edge $(v,p,v') \in E(G_P)$ if there exists a path in G from v to v' with maximum priority p and (whose projection on the mean payoff component is) rejected by $\mathscr{A}_{MP}$.

    We claim that G_P satisfies $\mathtt{Parity}_d$. Assume towards contradiction that G_P contains an odd cycle C_P. For each edge in this cycle there is a corresponding path rejected by $\mathscr{A}_{MP}$ with the same maximum priority. Putting these paths together yields an odd cycle C, of maximal priority p, in G whose projection on the mean payoff component is rejected by $\mathscr{A}_{MP}$. Since $\mathtt{MeanPayoff}_N^{|n} \subseteq L(\mathscr{A}_{MP})$ and as we have shown, G_{MP} satisfies $\mathtt{MeanPayoff}_N$, the projection of C on the mean payoff component is not in G_{MP}, so there exists an edge $(v,(p',w),v')$ in C such that (v,w,v') is not in $E(G_{MP})$. This implies that p' is even, so in particular $p' < p$, and $(v,(p',w),v')$ is contained in a negative cycle C' in G with maximum priority p'. Combining the odd cycle C followed by sufficiently many iterations of the negative cycle C' yields a path in G, with negative weight, and maximal priority p which is odd, a contradiction.

  Let π an infinite path in G, we show that π is accepted by $\mathscr{A}_{P\vee MP}$. Let us consider first only the mean payoff component: we run π repeatedly over $\mathscr{A}_{MP}$, and distinguish two cases.

  - If there are finitely many resets, let us write $\pi = \pi_1\pi_2\dots\pi_k\pi'$ where $\pi_1,\dots,\pi_k$ are paths rejected by $\mathscr{A}_{MP}$ with proper prefixes accepted by $\mathscr{A}_{MP}$ and π' is accepted by $\mathscr{A}_{MP}$. To show that π is accepted by $\mathscr{A}_{P\vee MP}$ we need to show that the automaton $\mathscr{A}_P$ accepts the word $p_1\dots p_k$ where p_i is the maximum priority appearing in π_i for $i \in [1,k]$. Indeed, $p_1\dots p_k$ is a path in G_P, which is a graph of n satisfying $\mathtt{Parity}_d$, so $\mathscr{A}_P$ accepts $p_1\dots p_k$.
  - If there are infinitely many resets, let us write $\pi = \pi_1\pi_2\dots$ where for each $i \in \mathbb{N}$, the path π_i is rejected by $\mathscr{A}_{MP}$ and its proper prefixes are not. To show that π is accepted by $\mathscr{A}_{P\vee MP}$ we need to show that the automaton $\mathscr{A}_P$ accepts the word $p_1p_2\dots$, which holds for the same reason as the other case: $p_1p_2\dots$ is a path in G_P.

□

3.2 The complexity of solving disjunctions of parity and mean payoff games using separating automata

The class of games with a disjunction of a parity and a mean payoff objective have been introduced in [3], with a twist: this paper studies the case of a conjunction instead of a disjunction, which is more natural in many applications. This is equivalent here since both parity and mean payoff objectives are dual, in other words if the objective of Eve is a disjunction of a parity and a mean payoff objective, then the objective of Adam is a conjunction of a parity and a mean payoff objective. Hence all existing results apply here with suitable changes. The reason why we consider the objective of the opponent is that it is positionally determined, which is the key assumption for using the separating automaton technology.

The state of the art for solving disjunctions of parity and mean payoff games is due to [8], which presents a pseudo-quasi-polynomial algorithm. We refer to [8] for references on previous studies for this class of games. As they explain, these games are logarithmic space equivalent to the same games replacing mean payoff by energy, and polynomial time equivalent to games with weights [14], extending games with costs [11].

Combining Theorem 6, Theorem 3, Theorem 4, and Theorem 1 yields the following result.

Theorem 7. *Let $n,d,N \in \mathbb{N}$. There exists an algorithm in the RAM model with word size $w = \log(n) + \log(N)$ for solving disjunctions of parity and mean payoff games with n vertices, m edges, weights in $[-N,N]$ and priorities in $[0,d]$ of time complexity*

$$O\left(md \cdot \underbrace{n \cdot \binom{\lceil \log(n) \rceil + d/2 - 1}{\lceil \log(n) \rceil}}_{\textit{Parity}} \cdot \underbrace{nN}_{\textit{Mean Payoff}}\right).$$

and space complexity $O(n)$.

Our algorithm is similar to the one constructed in [8]: they are both value iteration algorithms (called progress measure lifting algorithm in [8]), combining the two value iteration algorithms for parity and mean payoff games. However, the set of values are not the same (our algorithm stores an additional priority) and the proofs are very different. Besides being much shorter, one advantage of our proof is that it works with abstract separating automata for both parity and mean payoff objectives, and shows how to combine them, whereas in [8] the proof is done from scratch, extending both proofs for the parity and mean payoff objectives.

4 Disjunction of mean payoff

We define the objective $\bigvee_{i \in [1,d]} \texttt{MeanPayoff}_N^i$ referred to as 'disjunction of mean payoff'. The set of colours is $[-N,N]^d \subseteq \mathbb{Z}^d$. For $w \in ([-N,N]^d)^\omega$ and $i \in [1,d]$ we write $w_i \in [-N,N]^\omega$ for the projection on the i-th component.

$$\bigvee_{i \in [1,d]} \texttt{MeanPayoff}_N^i = \left\{ w \in ([-N,N]^d)^\omega : \exists i \in [1,d], w_i \in \texttt{MeanPayoff}_N \right\}.$$

We refer to Subsection 4.3 for a discussion on existing results.

Theorem 8 (Follows from [13] as observed in [15])**.** *Disjunctions of mean payoff objectives are prefix independent and positionally determined.*

4.1 A general reduction of separating automata for strongly connected graphs

The first idea is very general, it roughly says that if we know how to construct separating automata for strongly connected graphs, then we can use them to construct separating automata for general graphs.

We say that a graph is strongly connected if for every pair of vertices there exists a path from one vertex to the other. Let us refine the notion of separating automata. We write $\Omega^{|n}_{\text{sc}}$ for the set of infinite sequences of colours that label paths from strongly connected graphs of size at most n satisfying Ω. Formally,

$$\Omega^{|n}_{\text{sc}} = \{\text{col}(\pi) \mid \pi \in \text{Path}_\infty(G), G \text{ is strongly connected, has size at most } n, \text{ and satisfies } \Omega\}.$$

Definition 5 (Separating automata for strongly connected graphs)**.** *An automaton $\mathscr{A}$ is (n,Ω)-separating for strongly connected graphs if $\Omega^{|n}_{sc} \subseteq L(\mathscr{A}) \subseteq \Omega$.*

Let us give a first construction, which we refine later. Let $\mathscr{A}_1 = (Q_1, q_{0,1}, \delta_1)$ and $\mathscr{A}_2 = (Q_2, q_{0,2}, \delta_2)$ two safety automata, we define their sequential product $\langle \mathscr{A}_1, \mathscr{A}_2 \rangle$ as follows: the set of states is $Q_1 \cup Q_2$, the initial state is $q_{0,1}$, and the transition function is

$$\delta(s,c) = \begin{cases} \delta_1(s,c) & \text{if } s \in Q_1 \text{ and } \delta_1(s,c) \text{ is defined,} \\ q_{0,2} & \text{if } s \in Q_1 \text{ and } \delta_1(s,c) \text{ is not defined,} \\ \delta_2(s,c) & \text{if } s \in Q_2 \text{ and } \delta_2(s,c) \text{ is defined.} \end{cases}$$

The sequential product is extended inductively: $\langle \mathscr{A}_1, \dots, \mathscr{A}_p \rangle = \langle \langle \mathscr{A}_1, \dots, \mathscr{A}_{p-1} \rangle, \mathscr{A}_p \rangle$.

Lemma 3. *Let Ω be a prefix independent objective, $n \in \mathbb{N}$, and $\mathscr{A}$ an (n,Ω)-separating automaton for strongly connected graphs. Then $\mathscr{A}^n = \underbrace{\langle \mathscr{A}, \dots, \mathscr{A} \rangle}_{n \text{ times}}$ is an (n,Ω)-separating automaton.*

Proof. We first show that $L(\mathscr{A}^n) \subseteq \Omega$. We note that any infinite run in $\mathscr{A}^n$ eventually remains in one copy of $\mathscr{A}$. Since $\mathscr{A}$ satisfies Ω and by prefix independence of Ω, this implies that the infinite path satisfies Ω, thus so does $\mathscr{A}^n$.

Let G be a graph satisfying Ω, we show that $\text{Path}_\infty(G) \subseteq L(\mathscr{A}^n)$. We decompose G into strongly connected components: let $G_1, \dots, G_p$ be the maximal strongly connected components in G indexed such that if there exists an edge from G_i to G_j then $i < j$. Then $V(G)$ is the disjoint union of the $V(G_i)$'s. Each G_i is a subgraph of G, and since G satisfies Ω then so does G_i. It follows that for each i we have $\text{Path}_\infty(G_i) \subseteq L(\mathscr{A})$.

Let us consider a path π in G, we show that π is accepted by $\mathscr{A}^n$. The proof is by induction on the number of strongly connected components that π traverses. If it traverses only one such component, say G_i, since $\text{Path}_\infty(G_i) \subseteq L(\mathscr{A})$ then indeed $\mathscr{A}$ accepts π, so a fortiori $\mathscr{A}^n$ accepts π. Otherwise, let G_i the first strongly connected component traversed by π. Since $\text{Path}_\infty(G_i) \subseteq L(\mathscr{A})$, the run on π remains in the first copy of $\mathscr{A}$ at least as long as π remains in G_i. If the run remains in the first copy of $\mathscr{A}$ forever, then π is accepted by $\mathscr{A}^n$. If not, the run jumps to the second component to read a suffix of π, which traverses one less strongly connected component, so by induction hypothesis this suffix is accepted by $\mathscr{A}^{n-1}$, hence π is accepted by $\mathscr{A}^n$. □

In the construction above we have used the fact that G decomposes into at most n strongly connected components of size n. We refine this argument: the total size of the strongly connected components is n. The issue we are facing in taking advantage of this observation is that the sequence of sizes is not known a priori, it depends on the graph. To address this we use a universal sequence. First, here a sequence means a finite sequence of non-negative integer, for instance $(5,2,3,3)$. The size of a sequence is the total sum of its elements, so $(5,2,3,3)$ has size 13. We say that $v = (v_1,\dots,v_k)$ embeds into $u = (u_1,\dots,u_{k'})$ if there exists an increasing function $f : [1,k] \to [1,k']$ such that for all $i \in [1,k]$, we have $v_i \leq u_{f(i)}$. For example $(5,2,3,3)$ embeds into $(4,6,1,2,4,1,3)$ but not in $(3,2,5,3,3)$. A sequence u is n-universal if all sequences of size at most n embed into u.

Let us define an n-universal sequence u_n, inductively on $n \in \mathbb{N}$. We set $u_0 = ()$ (the empty sequence), $u_1 = (1)$, and u_n is the concatenation of $u_{\lfloor n/2 \rfloor}$ with the singleton sequence (n) followed by $u_{n-1-\lfloor n/2 \rfloor}$. Writing $+$ for concatenation, the definition reads $u_n = u_{\lfloor n/2 \rfloor} + (n) + u_{n-1-\lfloor n/2 \rfloor}$ Let us write the first sequences:

$$u_2 = (1,2), \quad u_3 = (1,3,1), \quad u_4 = (1,2,4,1), \quad u_5 = (1,2,5,1,2), \quad u_6 = (1,3,1,6,1,2),\dots$$

Lemma 4. *The sequence u_n is n-universal and has size $O(n\log(n))$.*

Proof. We proceed by induction on n. The case $n = 0$ is clear, let us assume that $n > 0$. Let $v = (v_1,\dots,v_k)$ be a sequence of size n, we show that v embeds into u_n. There exists a unique $p \in [1,k]$ such that $(v_1,\dots,v_{p-1})$ has size smaller than or equal to $\lfloor n/2 \rfloor$ and $(v_1,\dots,v_p)$ has size larger than $\lfloor n/2 \rfloor$. This implies that $(v_{p+1},\dots,v_k)$ has size at most $n-1-\lfloor n/2 \rfloor$. By induction hypothesis $(v_1,\dots,v_{p-1})$ embeds into $u_{\lfloor n/2 \rfloor}$ and $(v_{p+1},\dots,v_k)$ embeds into $u_{n-1-\lfloor n/2 \rfloor}$, so v embeds into u_n.

The recurrence on size is $|u_n| = |u_{\lfloor n/2 \rfloor}| + n + |u_{n-1-\lfloor n/2 \rfloor}|$. Solving it shows that $|u_n|$ is bounded by $O(n\log(n))$. □

We now use the universal sequence to improve on Lemma 3.

Lemma 5. *Let Ω be a positionally determined prefix independent objective and $n \in \mathbb{N}$. For each $k \in [1,n]$, let $\mathscr{A}_k$ be a (k,Ω)-separating automaton for strongly connected graphs. Let us write $u_n = (x_1,\dots,x_k)$, then $\mathscr{A}(u_n) = \langle \mathscr{A}_{x_1},\dots,\mathscr{A}_{x_k} \rangle$ is an (n,Ω)-separating automaton.*

Proof. We follow the same lines as for Lemma 3, in particular the same argument implies that $\mathscr{A}(u_n)$ satisfies Ω.

Let G be a graph satisfying Ω, we show that $\mathrm{Path}_\infty(G) \subseteq L(\mathscr{A}(u_n))$. We decompose G into strongly connected components as before. Let us write $v = (|V(G_1)|,\dots,|V(G_p)|)$ the sequence of sizes of the components. The sequence v has size at most n, implying that v embeds into u_n: there exists an increasing function $f : [1,p] \to [1,|u_n|]$ such that for all $i \in [1,p]$ we have $|V(G_i)| \leq u_{f(i)}$. It follows that for each $i \in [1,p]$, we have $\mathrm{Path}_\infty(G_i) \subseteq L(\mathscr{A}_{f(i)})$.

Let us consider a path π in G, we show that π is accepted by $\mathscr{A}(u_n)$. The proof is by induction on the number of strongly connected components that π traverses.

The base case is if π traverses only one such component, say G_i. This implies that the run of π on $\mathscr{A}(u_n)$ either remains in the first $f(i)-1$ copies, or reaches the $f(i)$ copy to read a suffix π' of π. In the latter case, since $\mathrm{Path}_\infty(G_i) \subseteq L(\mathscr{A}_{f(i)})$ and π' also remains in G_i, then $\mathscr{A}_{f(i)}$ accepts π'. In both cases π is accepted by $\mathscr{A}(u_n)$.

Otherwise, let G_i the first strongly connected component traversed by π. The run of π on $\mathscr{A}(u_n)$ either remains in the first $f(i)-1$ copies, or reaches the $f(i)$ copy to read a suffix π' of π. Since $\mathrm{Path}_\infty(G_i) \subseteq L(\mathscr{A}_{f(i)})$, this run remains in the $f(i)$ copy as long as π' remains in G_i. This can either hold forever, in which case π is accepted by $\mathscr{A}(u_n)$, or eventually the run jumps to the $f(i)+1$ copy to read a suffix π'' of π' (hence of π). In the latter case, since π'' traverses one less strongly connected component by induction hypothesis π'' is accepted using the copies from $f(i)+1$ of $\mathscr{A}(u_n)$. Thus π is accepted by $\mathscr{A}(u_n)$. □

To appreciate the improvement of Lemma 5 over Lemma 3, let us consider the case where the (k,Ω)-separating automaton $\mathscr{A}_k$ for strongly connected graphs has size αk, where α does not depend on k (see Theorem 4 for an example of such a case). Then Lemma 3 yields an (n,Ω)-separating automaton of size αn^2 while Lemma 5 brings it down to $O(\alpha n \log(n))$.

4.2 Separating automata for disjunctions of mean payoff objectives

We state in the following theorem an upper bound on the construction of separating automata for disjunctions of mean payoff objectives.

Theorem 9. *Let $n,d,N \in \mathbb{N}$. There exists an $(n, \bigvee_{i\in[1,d]} \mathtt{MeanPayoff}^i_N)$-separating automaton of size $O(n\log(n)\cdot dN)$.*

As suggested by the previous subsection, we will start by constructing $(n, \bigvee_{i\in[1,d]} \mathtt{MeanPayoff}^i_N)$-separating automata for strongly connected components. The following result shows a decomposition property.

Lemma 6. *Let G be a strongly connected graph and $N \in \mathbb{N}$. If G satisfies $\bigvee_{i\in[1,d]} \mathtt{MeanPayoff}^i_N$ then there exists $i \in [1,d]$ such that G satisfies $\mathtt{MeanPayoff}^i_N$.*

Proof. We prove the contrapositive property: assume that for all $i \in [1,d]$ the graph does not satisfy $\mathtt{MeanPayoff}^i_N$, implying that for each $i \in [1,d]$ there exists a negative cycle C_i around some vertex v_i. By iterating each cycle an increasing number of times, we construct a path in G which does not satisfy $\bigvee_i \mathtt{MeanPayoff}^i_N$.

To make this statement formal, we use the following property: for any $i \in [1,d]$, any finite path π can be extended to a finite path $\pi\pi'$ with weight less than -1 on the i^{th} component. This is achieved simply by first going to v_i using strong connectedness, then iterating through cycle C_i a sufficient number of times.

We then apply this process repeatedly and in a cyclic way over $i \in [1,d]$ to construct an infinite path such that for each $i \in [1,d]$, infinitely many times the i-th component is less than -1. This produces a path which does not satisfy $\bigvee_{i\in[1,d]} \mathtt{MeanPayoff}^i_N$, a contradiction. $\square$

Corollary 1. *Let $n,N \in \mathbb{N}$ and $\mathscr{A}$ be an $(n,\mathtt{MeanPayoff}_N)$-separating automaton for strongly connected graphs. We construct $d\cdot\mathscr{A}$ the disjoint union of d copies of $\mathscr{A}$, where the i-th copy reads the i-th component. Then $d\cdot\mathscr{A}$ is an $(n, \bigvee_{i\in[1,d]} \mathtt{MeanPayoff}^i_N)$-separating automaton for strongly connected graphs.*

We can now prove Theorem 9. Thanks to Theorem 4, there exists an $(n,\mathtt{MeanPayoff}_N)$-separating automaton of size nN. Thanks to Corollary 1, this implies an $(n, \bigvee_{i\in[1,d]} \mathtt{MeanPayoff}^i_N)$-separating automaton for strongly connected graphs of size ndN. Now Lemma 5 yields an $(n, \bigvee_i \mathtt{MeanPayoff}^i_N)$-separating automaton of size $O(n\log(n)\cdot dN)$.

4.3 The complexity of solving disjunctions of mean payoff games using separating automata

As in the previous case disjunction of mean payoff games were studied focussing on the objective of the opponent, who has a conjunction of mean payoff objectives to satisfy [15]. Let us note here that there are actually two variants of the mean payoff objective: using infimum limit or supremum limit. In many cases the two variants are equivalent: this is the case when considering mean payoff games or disjunctions of parity and mean payoff games. However, this is not true anymore for disjunctions of mean payoff objectives, as explained in [15]. Our constructions and results apply using the infimum limit, and do not extend to the supremum limit.

The main result related to disjunction of mean payoff games in [15] (Theorem 6) is that the problem is in NP$\cap$coNP and can be solved in time $O(m \cdot n^2 \cdot d \cdot N)$. (Note that since we consider the dual objectives, the infimum limit becomes a supremum limit in [15].)

Combining Theorem 9, Theorem 4, and Theorem 1 yields the following result.

Corollary 2. *Let $n,m,d,N \in \mathbb{N}$. There exists an algorithm in the RAM model with word size $w = \log(n) + \log(N) + \log(d)$ for solving disjunctions of mean payoff games with weights in $[-N,N]$ with time complexity $O(m \cdot n\log(n) \cdot d \cdot N)$ and space complexity $O(n)$.*

Note that for the choice of word size, the two tasks for manipulating separating automata, namely computing $\delta(q,w)$ and checking whether $q \leq q'$ are indeed unitary operations as they manipulate numbers of order nN.

Conclusions

The conceptual contribution of this paper is to show that the notion of separating automata is a powerful tool for constructing algorithms. We illustrated this point with two applications, constructing algorithms for two classes of objectives. A key appeal of our approach is that we assumed the existence of separating automata both for parity and for mean payoff objectives and used them as black boxes to construct separating automata for objectives combining them.

It is tempting to use our approach for constructing separating automata for disjunctions of parity objectives. However, solving games with disjunctions of two parity objectives is NP-complete [4], hence we cannot hope for subexponential separating automata.

In this paper we offered upper bounds on the sizes of separating automata for two classes of objectives. We leave open the questions of proving (matching) lower bounds, which would indicate that this approach cannot yield better algorithms in these cases.

References

[1] Mikołaj Bojańczyk & Wojciech Czerwiński (2018): *An Automata Toolbox*. Available at `https://www.mimuw.edu.pl/~bojan/papers/toolbox-reduced-feb6.pdf`.

[2] Cristian S. Calude, Sanjay Jain, Bakhadyr Khoussainov, Wei Li & Frank Stephan (2017): *Deciding parity games in quasipolynomial time*. In: *STOC*, pp. 252–263, doi:10.1145/3055399.3055409.

[3] Krishnendu Chatterjee, Thomas A. Henzinger & Marcin Jurdziński (2005): *Mean-Payoff Parity Games*. In: *LICS*, pp. 178–187, doi:10.1109/LICS.2005.26.

[4] Krishnendu Chatterjee, Thomas A. Henzinger & Nir Piterman (2007): *Generalized Parity Games*. In Helmut Seidl, editor: *FoSSaCS*, *Lecture Notes in Computer Science* 4423, Springer, pp. 153–167, doi:10.1007/978-3-540-71389-0_12.

[5] Thomas Colcombet & Nathanaël Fijalkow (2018): *Parity games and universal graphs*. *CoRR* abs/1810.05106. Available at `https://arxiv.org/abs/1810.05106`.

[6] Thomas Colcombet & Nathanaël Fijalkow (2019): *Universal Graphs and Good for Games Automata: New Tools for Infinite Duration Games*. In: *FoSSaCS*, doi:10.1007/978-3-030-17127-8_1.

[7] Wojciech Czerwiński, Laure Daviaud, Nathanaël Fijalkow, Marcin Jurdziński, Ranko Lazić & Paweł Parys (2019): *Universal trees grow inside separating automata: Quasi-polynomial lower bounds for parity games*. In: *SODA*, doi:10.1137/1.9781611975482.142.

[8] Laure Daviaud, Marcin Jurdziński & Ranko Lazić (2018): *A pseudo-quasi-polynomial algorithm for mean-payoff parity games*. In: *LICS*, doi:10.1145/3209108.3209162.

[9] Nathanaël Fijalkow (2018): *An Optimal Value Iteration Algorithm for Parity Games*. *CoRR* abs/1801.09618. Available at `https://arxiv.org/abs/1801.09618`.

[10] Nathanaël Fijalkow, Paweł Gawrychowski & Pierre Ohlmann (2020): *Value Iteration Using Universal Graphs and the Complexity of Mean Payoff Games*. In: *MFCS*, doi:10.4230/LIPIcs.MFCS.2020.34.

[11] Nathanaël Fijalkow & Martin Zimmermann (2014): *Parity and Streett Games with Costs*. *Logical Methods in Computer Science* 10(2), doi:10.2168/LMCS-10(2:14)2014.

[12] Marcin Jurdziński & Ranko Lazić (2017): *Succinct progress measures for solving parity games*. In: *LICS*, doi:10.1109/LICS.2017.8005092.

[13] Eryk Kopczyński (2006): *Half-Positional Determinacy of Infinite Games*. In: *ICALP*, pp. 336–347, doi:10.1007/11787006_29.

[14] Sven Schewe, Alexander Weinert & Martin Zimmermann (2019): *Parity Games with Weights*. *Logical Methods in Computer Science* 15(3), doi:10.23638/LMCS-15(3:20)2019.

[15] Yaron Velner, Krishnendu Chatterjee, Laurent Doyen, Thomas A. Henzinger, Alexander M. Rabinovich & Jean-François Raskin (2015): *The complexity of multi-mean-payoff and multi-energy games*. *Information and Computation* 241, pp. 177–196, doi:10.1016/j.ic.2015.03.001.