City Research Online

City St George's, University of London

Citation: Daviaud, L., Jurdziński, M. & Lazić, R. (2018). A pseudo-quasi-polynomial algorithm for solving mean-payoff parity games. In: LICS '18 Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science. LICS, 2018. (pp. 325-334). New York, NY: ACM. ISBN 978-1-4503-5583-4

This is the accepted version of the paper.

This version of the publication may differ from the final published version. To cite this item please consult the publisher's version.

Permanent repository link: https://openaccess.city.ac.uk/id/eprint/21289/

Copyright and Reuse: Copyright and Moral Rights remain with the author(s) and/or copyright holders. Copies of full items can be used for personal research or study, educational, or not-for-profit purposes without prior permission or charge, unless otherwise indicated, provided that the authors, title and full bibliographic details are credited, a hyperlink and/or URL is given for the original metadata page and the content is not changed in any way. For full details of reuse please refer to City Research Online policy.

City Research Online: http://openaccess.city.ac.uk/ publications@citystgeorges.ac.uk

A pseudo-quasi-polynomial algorithm for mean-payoff parity games

Laure Daviaud Marcin Jurdziński Ranko Lazić
DIMAP, Department of Computer Science
University of Warwick
UK

Abstract

In a mean-payoff parity game, one of the two players aims both to achieve a qualitative parity objective and to minimize a quantitative long-term average of payoffs (aka. mean payoff). The game is zero-sum and hence the aim of the other player is to either foil the parity objective or to maximize the mean payoff.

Our main technical result is a pseudo-quasi-polynomial algorithm for solving mean-payoff parity games. All algorithms for the problem that have been developed for over a decade have a pseudo-polynomial and an exponential factors in their running times; in the running time of our algorithm the latter is replaced with a quasi-polynomial one. By the results of Chatterjee and Doyen (2012) and of Schewe, Weinert, and Zimmermann (2018), our main technical result implies that there are pseudo-quasi-polynomial algorithms for solving parity energy games and for solving parity games with weights.

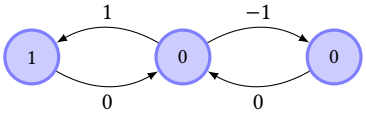
Our main conceptual contributions are the definitions of strategy decompositions for both players, and a notion of progress measures for mean-payoff parity games that generalizes both parity and energy progress measures. The former provides normal forms for and succinct representations of winning strategies, and the latter enables the application to mean-payoff parity games of the order-theoretic machinery that underpins a recent quasi-polynomial algorithm for solving parity games.

1 Introduction

A motivation to study zero-sum two-player games on graphs comes from automata theory and logic, where they have been used as a robust theoretical tool, for example, for streamlining of the initially notoriously complex proofs of Rabin's theorems on the complementation of automata on infinite trees and the decidability of the monadic second-order logic on infinite trees [16, 24], and for the development of the related theory of logics with fixpoint operators [12]. More practical motivations come from model checking and automated controller synthesis, where they serve as a clean combinatorial model for the study of the computational complexity and algorithmic techniques for model checking [13], and for the automated synthesis of correct-by-design controllers [23]. There is a rich literature on closely related "dynamic games" in the classical game theory and AI literatures reaching back to 1950's, and games on graphs are also relevant to complexity theory [9] and to competitive ratio analysis of online algorithms [25].

1.1 Mean-payoff parity games

A *mean-payoff parity* game is played by two players—Con and Dis—on a directed graph. From the starting vertex, the players keep following edges of the graph forever, thus forming an infinite path. The set of vertices is partitioned into those owned by Con and those owned by Dis, and it is the owner of the current vertex who picks which outgoing edge to follow to the next current vertex. Who is declared the winner of an infinite path formed by such interaction is determined by the labels of vertices and edges encountered on the path. Every vertex is labelled by a positive integer called its *priority* and every edge is labelled by an integer called its *cost*. The former are used to define the *parity* condition: the highest priority that occurs infinitely many times is *odd*; and the latter are used to define the (zero-threshold) *mean-payoff* condition: the (lim-sup) long-run average of the costs is negative. If both the parity and the mean-payoff conditions hold then Con is declared the winner, and otherwise Dis is. In the following picture, if Dis owns the vertex in the middle then she wins the game (with a positional strategy): she can for example always go to the left whenever she is in the middle vertex and this way achieve the positive mean payoff $1/2$. Conversely, if Con owns the middle vertex then he wins the game. He can choose to go infinitely often to the left and see priority 1—in order to fulfill the parity condition—and immediately after each visit to the left, to go to the right a sufficient number of times—so as to make the mean-payoff negative. Note that a winning strategy for Con is not positional.



Throughout the paper, we write V and E for the sets of vertices and directed edges in a mean-payoff parity game graph, $\pi(v)$ for the priority of a vertex $v \in V$, and $c(v,u)$ for the cost of an edge $(v,u) \in E$. Vertex priorities are positive integers no larger than d, which we assume throughout the paper to be a positive even integer, edge costs are integers whose absolute value does not exceed the positive integer C, and we write n and m for the numbers of vertices and edges in the graph, respectively.

Several variants of the algorithmic problem of *solving mean-payoff parity games* have been considered in the literature. The input always includes a game graph as described above. The *value* of (a vertex in) a mean-payoff parity game is defined as ∞ if Con does not have a winning strategy for the parity condition, and otherwise the smallest mean payoff that Con can secure while playing so as to satisfy the parity condition. (Note that the paper that introduced mean-payoff parity games [7] defined Con to be the maximizer and not, as we do, the minimizer of the mean payoff. The two definitions are straightforwardly inter-reducible; the choice we made allows for a better alignment of our key notion of a mean-payoff parity progress measure with the literature on energy progress measures [2].) The *value problem* is to compute the value of every vertex. The *threshold problem* is, given an additional (rational) number θ as a part of the input, to compute the set of

, ,
.

vertices with finite value (strictly) less than θ. (Note that a value of a vertex is not finite, i.e., it is ∞, if and only if Con does not have a winning strategy for his parity condition, which can be checked in quasi-polynomial time [3, 19].) In the *zero-threshold problem* the threshold number θ is assumed to be 0.

As Chatterjee et al. [6, Theorem 10] have shown, the threshold problem can be used to solve the value problem at the cost of increasing the running time by the modest $O(n \cdot \log(nC))$ multiplicative term. Their result, together with a routine linear-time reduction from the threshold problem to the zero-threshold problem (subtract θ from costs of all edges), motivate us to focus on solving the zero-threshold problem in this paper. For brevity, we will henceforth write "mean-payoff condition" instead of "zero-threshold mean-payoff condition".

The roles of the two players in a mean-payoff parity game are not symmetric for several reasons. One is that Con aims to satisfy a *conjunction* of the parity condition and of the mean-payoff condition, while the goal of Dis is to satisfy a *disjunction* of the negated conditions. The other one is that negations of the parity condition and of the mean-payoff condition are not literally the parity and the mean-payoff conditions, respectively: the negation of the parity condition swaps the roles of even and odd, and the negation of the strict ("less than") mean-payoff condition is non-strict ("at least"). The former asymmetry (conjunction vs disjunction) is material and our treatments of strategy construction for players Con and Dis differ substantially, but the latter are technically benign. The discussion above implies that the goal of player Dis is to either satisfy the parity condition in which the highest priority that occurs infinitely many times is even, or to satisfy the "at least" zero-threshold mean-payoff condition.

1.2 Related work

Mean-payoff games have been studied since 1960's and there is a rich body of work on them in the stochastic games literature. We selectively mention the positional determinacy result of Ehrenfeucht and Mycielski [11] (i.e., that positional optimal strategies exist for both players), and the work of Zwick and Paterson [25], who pointed out that positional determinacy implies that deciding the winner in mean-payoff games is both in NP and in co-NP, and gave a pseudo-polynomial algorithm for computing values in mean-payoff games that runs in time $O(mn^3C)$. Brim et al. [2] introduced energy progress measures as natural witnesses for winning strategies in closely related energy games, they developed a lifting algorithm to compute the least energy progress measures, and they observed that this leads to an algorithm for computing values in mean-payoff games whose running time is $O(mn^2C \cdot \log(nC))$, which is better than the algorithm of Zwick and Paterson [25] if $C = 2^{o(n)}$. Comin and Rizzi [8] have further refined the usage of the lifting algorithm for energy games achieving running time $O(mn^2C)$.

Parity games have been studied in the theory of automata on infinite trees, fixpoint logics, and in verification and synthesis since early 1990's [12, 13]. Very selectively, we mention early and influential recursive algorithms by McNaughton [21] and by Zielonka [24], the running times of which are $O(n^{d+O(1)})$. The breakthrough result of Calude et al. [3] gave the first algorithm that achieved an $n^{o(d)}$ running time. Its running time is polynomial $O(n^5)$ if $d \leq \log n$ and quasipolynomial $O(n^{\lg d+6})$ in general. (Throughout the paper, we write $\lg x$ to denote $\log_2 x$, and we write $\log x$ when the base of the logarithm is moot.) Note that Calude et al.'s polynomial bound for $d \leq \log n$ implies that parity games are FPT (fixed parameter tractable) when the number d of distinct vertex priorities is the parameter. Further analysis by Jurdziński and Lazić [19] established that running times $O(mn^{2.38})$ for $d \leq \lg n$, and $O(dmn^{\lg(d/\lg n)+1.45})$ for $d = \omega(\lg n)$, can be achieved using their succinct progress measures, and Fearnley et al. [14] obtained similar results by refining the technique and the analysis of Calude et al. [3]. Existence of polynomial-time algorithms for solving parity games and for solving mean-payoff games are fundamental long-standing open problems [13, 17, 25].

Mean-payoff parity games have been introduced by Chatterjee et al. [7] as a proof of concept in developing algorithmic techniques for solving games (and hence for controller synthesis) which combine qualitative (functional) and quantitative (performance) objectives. Their algorithm for the value problem is inspired by the recursive algorithms of McNaughton [21] and Zielonka [24] for parity games, from which its running time acquires the exponential dependence $mn^{d+O(1)}C$ on the number of vertex priorities. Chatterjee and Doyen [4] have simplified the approach by considering energy parity games first, achieving running time $O(dmn^{d+4}C)$ for the threshold problem, which was further improved by Bouyer et al. [1] to $O(mn^{d+2}C)$ for the value problem. Finally, Chatterjee et al. [6] have achieved the running time $O(mn^dC\log(nC))$ for the value problem, but their key original technical results are for the two special cases of mean-payoff parity games that allow only two distinct vertex priorities, for which they achieve running time $O(mnC)$ for the threshold problem, by using amortized analysis techniques from dynamic algorithms. Note that none of those algorithms escapes the exponential dependence on the number of distinct vertex priorities, simply because they all follow the recursive structure of the algorithms by McNaughton [21] and by Zielonka [24].

Other quantitative extensions of parity games have been considered; for example, Fijalkow and Zimmermann [15] introduced *parity games with costs*, and Schewe, Weinert, and Zimmermann [22] generalized those to *parity games with weights*. Chatterjee and Doyen [4] have proved that the problem of deciding the winner in mean-payoff parity games is log-space equivalent to the problem of deciding the winner in energy parity games, and Schewe et al. [22] have proved that the latter is polynomial-time equivalent to the problem of deciding the winner in parity games with weights. It follows that the three problems, of deciding the winner in mean-payoff parity games, in energy parity games, and in parity games with weights, respectively, are polynomial-time equivalent.

1.3 Our contributions

Our main technical result is the first pseudo-quasi-polynomial algorithm for solving mean-payoff parity games. More specifically, we prove that the threshold problem can be solved in pseudo-polynomial time $mn^{2+o(1)}C$ for $d = o(\log n)$, in pseudo-polynomial time $mn^{O(1)}C$ if $d = O(\log n)$ (where the constant in the exponent of n depends logarithmically on the constant hidden in the big-Oh expression $O(\log n)$), and in pseudo-quasi-polynomial time $O(dmn^{\lg(d/\lg n)+2.45}C)$ if $d = \omega(\log n)$. By [6, Theorem 10], we obtain running times for solving the value problem that are obtained from the ones above by multiplying them by the $O(n\log(nC))$ term.

Our key conceptual contributions are the notions of strategy decompositions for both players in mean-payoff parity games, and of mean-payoff parity progress measures. The former explicitly reveal the underlying strategy structure of winning sets for both players, and they provide normal forms and succinct representations for winning strategies. The latter provide an alternative form of a witness and a normal form of winning strategies for player Dis, which make explicit the order-theoretic structures that underpin the original progress measure lifting algorithms for parity [18] and energy games [2], respectively, as well as the recent quasi-polynomial succinct progress measure lifting algorithm for parity games [19]. The proofs of existence of strategy decompositions follow the well-beaten track of using McNaughton-Zielonka-like inductive arguments, and existence of progress measures that witness winning strategies for Dis is established by extracting them from strategy decompositions for Dis.

Our notion of mean-payoff parity progress measures combines features of parity and energy progress measures, respectively. Crucially, our mean-payoff progress measures inherit the ordered tree structure from parity progress measures, and the additional numerical labels of vertices (that capture the energy progress measure aspects) do not interfere substantially with it. This allows us to directly apply the combinatorial ordered tree coding result by Jurdziński and Lazić [19], which limits the search space in which the witnesses are sought by the lifting procedure to a pseudo-quasi-polynomial size, yielding our main result. The order-theoretic properties that the lifting procedure relies on naturally imply the existence of the least (in an appropriate order-theoretic sense) progress measure, from which a positional winning strategy for Dis on her winning set can be easily extracted.

In order to synthesize a strategy decomposition—and hence a winning strategy—for Con in pseudo-quasi-polynomial time, we take a different approach. Progress measures for games typically yield positional winning strategies for the relevant player [2, 18, 20], but optimal strategies for Con in mean-payoff parity games may require infinite memory [7]. That motivates us to forgo attempting to pin a notion of progress measures to witness winning strategies for Con. We argue, instead, that a McNaughton-Zielonka-style recursive procedure can be modified to run in pseudo-quasi-polynomial time and produce a strategy decomposition of Con's winning set. The key insight is to avoid invoking some of the recursive calls, and instead to replace them by invocations of the pseudo-quasi-polynomial lifting procedure for Dis, merely to compute the winning set for Dis—and hence also for Con, because by determinacy Con has a winning strategy whenever Dis does not. As a result, each invocation of the recursive procedure only makes recursive calls on disjoint subgames, which makes it perform only a polynomial number of steps other than invocations of the lifting procedure, overall yielding a pseudo-quasi-polynomial algorithm.

Note that our pseudo-quasi-polynomial algorithm for mean-payoff parity games can be used to solve energy parity games and parity games with weights in pseudo-quasi-polynomial time, because deciding the winner in the latter two classes of games is polynomial-time equivalent to deciding the winner in mean-payoff games by the results of Chatterjee and Doyen [4] and Schewe et al. [22], respectively.

Organisation of the paper. In Section 2, we define strategy decompositions for Dis and Con, and we prove that they exist if and only if the respective player has a winning strategy. In Section 3, we define progress measures for Dis, and we prove that such a progress measure exists if and only if Dis has a strategy decomposition. In Section 4, we give a pseudo-quasi-polynomial lifting algorithm for computing the least progress measure, from which a strategy decomposition for Dis of her winning set, and the winning set for Con, can be derived. In Section 5, we show how to also compute a strategy decomposition for Con on his winning set in pseudo-quasi-polynomial time, using the lifting procedure to speed up a NcNaughton-Zielonka-style recursive procedure.

2 Strategy decompositions

In this section we introduce our first key concept of *strategy decompositions* for each of the two players. They are hierarchically defined objects, of size polynomial in the number of vertices in the game graph, that witness existence of winning strategies for each of the two players on their winning sets. Such decompositions are implicit in earlier literature, in particular in algorithms for mean-payoff parity games [1, 4, 6, 7] that follow the recursive logic of McNaughton's [21] and Zielonka's [24] algorithms for parity games. We make them explicit because we belive that it provides conceptual clarity and technical advantages. Strategy decompositions pinpoint the recursive strategic structure of the winning sets in mean-payoff parity games (and, by specialization, in parity games too), which may provide valuable insights for future work on the subject. What they allow us to do in this work is to streamline the proof that the other key concept we introduce—mean-payoff parity progress measures—witness existence of winning strategies for Dis.

We define the notions of strategy decompositions for Dis and for Con, then in Lemmas 2.1 and 2.2 we prove that the decompositions naturally yield winning strategies for the corresponding players, and finally in Lemma 2.3 we establish that in every mean-payoff game, both players have strategy decompositions of their winning sets. The proofs of all three lemmas mostly use well-known inductive McNaughton-Zielonka-type arguments that should be familiar to anyone who is conversant in the existing literature on mean-payoff parity games. We wish to think that for a curious non-expert, this section offers a streamlined and self-contained exposition of the key algorithmic ideas behind earlier works on mean-payoff parity games [1, 4, 7].

2.1 Preliminaries

Notions of strategies, positional strategies, plays, plays consistent with a strategy, winning strategies, winning sets, reachability strategies, traps, mean payoff, etc., are defined in the usual way. We forgo tediously repeating the definitions of those common and routine concepts, referring a non-expert but interested reader to consult the (typically one-page) Preliminaries or Definitions sections of any of the previously published papers on mean-payoff parity games [1, 4, 6, 7]. One notable difference between our set-up and those found in the above-mentioned papers is that for an infinite sequence of numbers $\langle c_1, c_2, c_3, \ldots \rangle$, we define its mean payoff to be $\limsup_{n\to\infty}(1/n)\cdot\sum_{i=1}^{n} c_i$, rather than the more common $\liminf_{n\to\infty}(1/n)\cdot\sum_{i=1}^{n} c_i$; this is because we chose Con to be the minimizer of the mean payoff, instead of the typical choice of making him the maximizer.

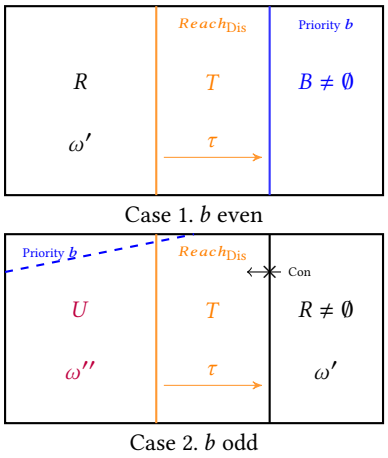


Figure 1. Strategy decompositions for Dis.

2.2 Strategy decompositions for Dis

Let $W \subseteq V$ be a subgame (*i.e.* a non empty induced subgraph of V with no deadend) in which the biggest vertex priority is b. We define strategy decompositions for Dis by induction on b and the size of W. We say that ω is a b-decomposition of W for Dis if the following conditions hold (pictured in Figure 1).

1. If b is even then $\omega = \big((R, \omega'), (T, \tau), B\big)$, such that:
   a. sets R, T, and $B \neq \emptyset$ are a partition of W;
   b. B is the set of vertices of the top priority b in W;
   c. τ is a positional reachability strategy for Dis from T to B in W;
   d. ω' is a b'-decomposition of R for Dis, where $b' < b$.
2. If b is odd then $\omega = \big((U, \omega''), (T, \tau), (R, \omega')\big)$, such that:
   a. sets U, T, and $R \neq \emptyset$ are a partition of W;
   b. ω' is either:
      i. a b'-decomposition of R for Dis, where $b' < b$; or
      ii. a positional strategy for Dis that is mean-payoff winning for her on R;
   c. τ is a positional reachability strategy for Dis from T to R in W;
   d. ω'' is a b''-decomposition of U for Dis, where $b'' \leq b$;
   e. R is a trap for Con in W.

We say that a subgame W has a strategy decomposition for Dis if it has a b-decomposition for some b. A heuristic, if somewhat non-standard, way to think about sets T and R in the above definition is that sets denoted by T are *transient* and sets denoted by R are *recurrent*. The meanings of those words here are different than in, say, Markov chains, and refer to strategic, rather than probabilistic, properties.

Given a strategy decomposition ω for Dis, we inductively define a positional strategy $\sigma(\omega)$ for Dis in the following way:

$$\sigma(\omega) = \begin{cases} \sigma(\omega') \cup \tau \cup \beta & \text{if } \omega = \big((R, \omega'), (T, \tau), B\big), \\ \sigma(\omega'') \cup \tau \cup \sigma(\omega') & \text{if } \omega = \big((U, \omega''), (T, \tau), (R, \omega')\big), \end{cases}$$

where β is an arbitrary positional strategy for Dis on B, and $\sigma(\omega') = \omega'$ in case 2(b)ii.

Lemma 2.1. *If ω is a strategy decomposition of W for Dis and W is a trap for Con, then $\sigma(\omega)$ is a positional winning strategy for Dis from every vertex in W.*

Proof. We proceed by induction on the number of vertices in W. The reasoning involved in the base cases (when $R = \emptyset$ or $U = \emptyset$) is analogous and simpler than in the inductive cases, hence we immediately proceed to the latter.

We consider two cases based on the parity of the biggest vertex priority b in W.

First, assume that b is even and let $\omega = \big((R, \omega'), (T, \tau), B\big)$ be a b-decomposition of W. We argue that every infinite play consistent with $\sigma(\omega)$ is winning for Dis. If it visits vertices in B infinitely many times then the parity condition for Dis is satisfied because b is the biggest vertex priority and it is even. Otherwise, it must be the case that the play visits vertices in $T \cup B$ only finitely many times, because visiting a vertex in T always leads in finitely many steps to visiting a vertex in B by following the reachability strategy τ. Therefore, eventually the play never leaves R and is consistent with strategy $\sigma(\omega')$, which is winning for Dis by the inductive hypothesis.

Next, assume that b is odd. Let $\omega = \big((U, \omega''), (T, \tau), (R, \omega')\big)$ be a b-decomposition. We argue that every infinite play consistent with $\sigma(\omega)$ is winning for Dis. If it visits $T \cup R$, then by following strategy τ, it eventually reaches and never leaves R (because R is a trap for Con), and hence it is winning for Dis because $\sigma(\omega')$ is a winning strategy for Dis by the inductive hypothesis, or by condition 2(b)ii. Otherwise, if such a play never visits $T \cup R$ then it is winning for Dis because $\sigma(\omega'')$ is a winning strategy for Dis by the inductive hypothesis. □

2.3 Strategy decompositions for Con

Let $W \subseteq V$ be a subgame in which the biggest vertex priority is b. We define strategy decompositions for Con by induction on b and the size of W. We say that ω is a b-decomposition of W for Con if the following conditions hold (pictured in Figure 2).

1. If b is odd then $\omega = \big((R, \omega'), (T, \tau), B, \lambda\big)$, such that:
   a. sets R, T, and $B \neq \emptyset$ are a partition of W;
   b. B is the set of vertices of priority b in W;
   c. τ is a positional reachability strategy for Con from T to B in W;
   d. ω' is a b'-decomposition of R for Con, where $b' < b$;
   e. λ is a positional strategy for Con that is mean-payoff winning for him on W.
2. If b is even then $\omega = \big((U, \omega''), (T, \tau), (R, \omega')\big)$, such that:
   a. sets U, T, and $R \neq \emptyset$ are a partition of W;
   b. ω' is a b'-decomposition of R for Con, where $b' < b$;
   c. τ is a positional reachability strategy for Con from T to R in W;
   d. ω'' is a b''-decomposition of U for Con, where $b'' \leq b$;
   e. R is a trap for Dis in W.

We say that a subgame has a strategy decomposition for Con if it has a b-decomposition for some b. Note that the definition is analogous to that of a strategy decomposition for Dis in most aspects, with the following differences:

- the roles of Dis and Con, and of even and odd, are swapped;
- the condition 2b is simplified;
- an extra component λ, and the condition 1e, are added.

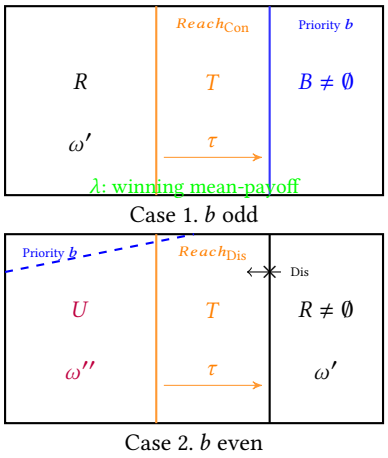


Figure 2. Strategy decompositions for Con.

Given a strategy decomposition ω for Con, we inductively define a strategy $\sigma(\omega)$ for Con in the following way:

- If b is odd and $\omega = ((R, \omega'), (T, \tau), B, \lambda)$, then the strategy proceeds in (possibly infinitely many) rounds. Round i, for $i = 1, 2, 3, \ldots$, involves the following steps:
  1. if starting in R, follow $\sigma(\omega')$ for as long as staying in R;
  2. if starting in T, or having arrived there from R, follow τ until B is reached;
  3. once B is reached, follow λ for $n + (2n + 3^n + 2)nC$ steps and proceed to round $i + 1$.
- If b is even and $\omega = ((U, \omega''), (T, \tau), (R, \omega'))$, then let:

$$\sigma(\omega) = \sigma(\omega'') \cup \tau \cup \sigma(\omega').$$

Lemma 2.2. *If ω is a strategy decomposition of W for Con and W is a trap for Dis, then $\sigma(\omega)$ is a winning strategy for Con from every vertex in W.*

Proof. We proceed by induction on the number of vertices in W, omitting the base cases (when $R = \emptyset$, or $U = \emptyset$, respectively), since they are analogous and simpler than the inductive cases.

We consider two cases based on the parity of b. First, assume that b is even and let $\omega = \big((U, \omega''), (T, \tau), (R, \omega')\big)$. Observe that a play consistent with $\sigma(\omega) = \sigma(\omega'') \cup \tau \cup \sigma(\omega')$ either never leaves U, or if it does then after a finite number of steps (following the reachability strategy τ) it enters R and then never leaves it because R is a trap for Dis. It follows that the play is winning for Con by the inductive hypothesis, because it is eventually either consistent with strategy $\sigma(\omega'')$ or with $\sigma(\omega')$.

Next, assume that b is odd. Let $\omega = \big((R, \omega'), (T, \tau), B, \lambda\big)$ be a b-decomposition. We argue that every infinite play consistent with $\sigma(\omega)$ is winning for Con. If it visits $T \cup R$ finitely many times then eventually it is consistent with strategy $\sigma(\omega')$ and hence it is winning for Con by the inductive hypothesis. Otherwise, it visits set B infinitely many times and hence it satisfies the parity condition for Con because b is the highest vertex priority and it is odd. We now claim that every play consistent with $\sigma(\omega)$ has a negative mean payoff, and hence also satisfies the mean-payoff condition for Con. For lack of space, we do not provide a detailed and rigorous argument here, but it can be found in the full version of this paper [10]. The main insight, however, is that in every round, either the strategy $\sigma(\omega')$ is played for long enough to—by the inductive hypothesis—contribute a negative average cost in step 1., or the combined costs encountered in steps 1. and 2. are bounded, and hence they are eventually overwhelmed by the negative averages achieved in arbitrarily long spells of playing a mean-payoff strategy λ (which, by positional determinacy of mean-payoff games [11], secures mean-payoff at most $-1/n$) in step 3. □

2.4 Existence of strategy decompositions

In the following lemma, we prove that every game can be partitioned into two sets of vertices, so that there is a strategy decomposition of one for Dis, and a strategy decomposition of the other one for Con. Those sets correspond to the winning sets for Dis and Con, respectively.

Lemma 2.3. *There is a partition W_{Dis} and W_{Con} of V, such that there is a strategy decomposition of W_{Dis} for Dis (provided $W_{\text{Dis}} \neq \emptyset$) and a strategy decomposition of W_{Con} for Con (provided $W_{\text{Con}} \neq \emptyset$).*

The proof of Lemma 2.3 can be found in the full version of the paper [10]. It follows the usual template of using a McNaughton-Zielonka inductive argument, as adapted to mean-payoff parity games by Chatterjee et al. [7], and then simplified for threshold mean-payoff parity games by Chatterjee et al. [6, Appendix C].

Observe that Lemmas 2.1, 2.2, and 2.3 form a self-contained argument to establish both determinacy of threshold mean-payoff parity games (from every vertex, one of the players has a winning strategy), and membership of the problem of deciding the winner both in NP and in co-NP. For the latter, it suffices to note that strategy decompositions can be described in a polynomial number of bits, and it can be routinely checked in small polynomial time whether a proposed strategy decomposition for either of the players satisfies all the conditions in the corresponding definition. The NP and co-NP membership has been first established by Chatterjee and Doyen [4]; we merely give an alternative proof.

Corollary 2.4 (Chatterjee and Doyen [4])**.** *The problem of deciding the winner in mean-payoff parity games is both in NP and in co-NP.*

3 Mean-payoff parity progress measures

In this section we introduce the other key concept—*mean-payoff parity progress measures*—that plays the critical role in achieving our main technical result—the first pseudo-quasi-polynomial algorithm for solving mean-payoff parity games. In Lemmas 3.1 and 3.2 we establish that mean-payoff parity progress measures witness existence of winning strategies for Dis, by providing explicit translations between them and strategy decompositions for Dis.

We stress that the purpose of introducing yet another concept of witnesses for winning strategies for Dis is to shift technical focus from highlighting the recursive strategic structure of winning sets in strategy decompositions, to an order theoretic formalization that makes the recursive structure be reflected in the concept of ordered trees. The order-theoretic formalization then allows us—in Section 4—to apply the combinatorial result on succinct coding of ordered trees by Jurdziński and Lazić [19], paving the way to the pseudo-quasi-polynomial algorithm.

3.1 The definition

A progress measure maps every vertex with an element of a linearly ordered set. Edges along which those elements decrease (for another

specific defined order) are called progressive, and an infinite path consisting only of progressive edges is winning for Dis. Then, we can derive a winning strategy for Dis if she can always follow a progressive edge and if Con has no other choice than following a progressive edge.

Recall the assumption that d—the upper bound on the vertex priorities—is even.

A *progress measurement* is a sequence $\big(\langle m_{d-1}, m_{d-3}, \ldots, m_\ell\rangle, e\big)$, where:

- ℓ is odd and $1 \leq \ell \leq d+1$ (note that if $\ell = d+1$ then $\langle m_{d-1}, m_{d-3}, \ldots, m_\ell\rangle$ is the empty sequence $\langle\rangle$);
- m_i is an element of a linearly ordered set (for simplicity, we write $\leq$ for the order relation), for each odd i, such that $\ell \leq i \leq d-1$;
- e is an integer such that $0 \leq e \leq nC$, or $e = \infty$.

A *progress labelling* (μ, φ) maps vertices to progress measurements in such a way that if vertex v is mapped to

$$\big(\mu(v), \varphi(v)\big) = \big(\langle m_{d-1}, m_{d-3}, \ldots, m_\ell\rangle, e\big)$$

then

- $\ell \geq \pi(v)$; and
- if $e = \infty$ then ℓ is the smallest odd integer such that $\ell \geq \pi(v)$.

For every priority p, $1 \leq p \leq d$, we obtain a *p-truncation* $\langle m_{d-1}, m_{d-3}, \ldots, m_\ell\rangle|_p$ of $\langle m_{d-1}, m_{d-3}, \ldots, m_\ell\rangle$, by removing the components corresponding to all odd priorities smaller than p. For example, if we fix $d = 8$ then we have $\langle a, b, c\rangle|_8 = \langle\rangle$, $\langle a, b, c\rangle|_6 = \langle a\rangle$, and $\langle a, b, c\rangle|_3 = \langle a, b, c\rangle|_2 = \langle a, b, c\rangle$. We compare sequences using the lexicographic order; for simplicity, and overloading notation, we write $\leq$ to denote it. For example, $\langle a\rangle < \langle a, b\rangle$, and $\langle a, b, c\rangle < \langle a, d\rangle$ if $b < d$.

Let (μ, φ) be a progress labelling. Observe that—by definition—$\mu(v)|_{\pi(v)} = \mu(v)$, for every vertex $v \in V$. We say that an edge $(v, u) \in E$ is *progressive* in (μ, φ) if:

1. $\mu(v) > \mu(u)|_{\pi(v)}$; or
2. $\mu(v) = \mu(u)|_{\pi(v)}$, $\pi(v)$ is even, and $\varphi(v) = \infty$; or
3. $\mu(v) = \mu(u)$, $\varphi(v) \neq \infty$, and $\varphi(v) + c(v, u) \geq \varphi(u)$.

We can represent tuples as nodes in a tree where the components of the tuple represent the branching directions in the tree to go from the root to the node. For example, a tuple $\langle a, b, c\rangle$ corresponds to the node reached from the root by first reaching the ath child of the root, then the bth child of this latter and finally the cth child of this one. This way, the notion of progressive edges can be seen on a tree as in Figure 3.

A progress labelling (μ, φ) is a *progress measure* if:

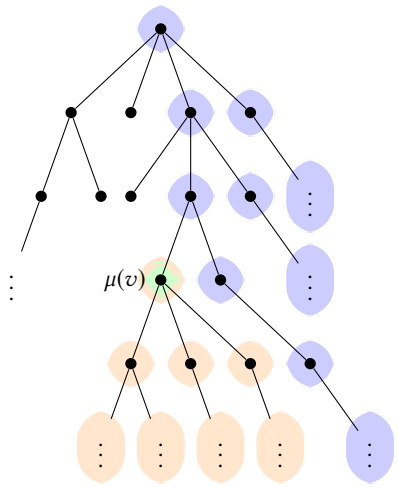
- for every vertex owned by Dis, there is at least one outgoing edge that is progressive in (μ, φ); and
- for every vertex owned by Con, all outgoing edges are progressive in (μ, φ).

In the next two sections, we prove that there is a strategy decomposition for Dis if and only is there is a progress measure.

3.2 From progress measures to strategy decompositions

Lemma 3.1. *If there is a progress measure then there is a strategy decomposition of V for Dis.*

In the proof we will use the following simple fact (see, for example, Brim et al. [2]): if all the edges in an infinite path are progressive and fulfill condition 3. of the definition, then the mean payoff of this path is non-negative (and thus winning for Dis).



The siblings are ordered according to the linear order $\leq$. The smallest child is on the right and the greatest on the left in the picture. An edge (v, u) is progressive if one of the three following conditions holds:

- condition 1 -

$\mu(u)$ is one of the blue node, *i.e.* above or on the right of $\mu(v)$.

- condition 2 -

$\pi(v)$ is even, $\varphi(v) = \infty$ and $\mu(u)$ is one of the orange node, *i.e.* belongs to the subtree rooted in $\mu(v)$.

- condition 3 -

$\mu(u) = \mu(v)$, $\varphi(u) \in \mathbb{Z}$ and $\varphi(v) + c(v, u) \geq \varphi(u)$.

Figure 3. Conditions for an edge to be progressive.

Proof. We proceed by induction on the number of distinct vertex priorities in the game graph. Let $b \leq d$ be the highest priority appearing in the game.

The base case is when b is the only vertex priority. If b is even, then by setting $B = V$ and $T = R = \emptyset$ we obtain a strategy decomposition of V for Dis. If b is odd, then an edge can only be progressive if it satisfies condition 3. of the definition of a progressive edge; hence the progress measure yields a positional strategy ω' for Dis that is mean-payoff winning for her on V. It follows that setting $R = V$, $T = U = \emptyset$, and ω' as above, we obtain a strategy decomposition of V for Dis.

Consider the inductive step now. First, suppose that b is even. Let B be the set of the vertices of priority b. Let T be the set of vertices from which Dis has a reachability strategy to B, τ be this positional strategy and let $R = V \setminus (B \cup T)$. Because, by construction, there is no edge from a vertex in R owned by Dis to a vertex in $B \cup T$, the progress measure on V gives also a progress measure on R when restricted to its vertices. Let ω be a b'-decomposition of R for Dis that exists by the inductive hypothesis. Note that $b' < b$ because the biggest priority in R is smaller than b. It follows that $\big((R, \omega), (T, \tau), B\big)$ is a strategy decomposition for Dis in V.

Suppose now that b is odd. Let R be the set of vertices labelled by the smallest tuple: $R = \{v \in V : \mu(v) \leq \mu(u) \text{ for all } u \in V\}$.

(If we pictured the tuples on a tree as in Figure 3, those would be the vertices that are mapped to the rightmost-top node in the tree among the nodes at least one vertex is mapped to.) Let R' be the subset of R of those vertices having a finite φ: $\{v \in R : \varphi(v) \neq \infty\}$.

Suppose first that $R' \neq \emptyset$. An edge going out from a vertex in R' can only be progressive if it fulfills condition 3. in the definition. It then has to go to a vertex of R' too. Thus, R' is a trap for Con, and Dis has a winning strategy ω' in R' for the mean-payoff game.

Let T be the set of vertices from which Dis has a strategy to reach R' and let τ this positional reachability strategy. Let $U = V \setminus (R' \cup T)$. Because, by construction, there is no edge from a vertex in U owned by Dis to a vertex in $R' \cup T$, then the progress measure on V gives also a progress measure on U when restricted to its vertices. We can then apply the inductive hypothesis and get ω a strategy decomposition of U for Dis. Note that $\big((U, \omega), (T, \tau), (R', \omega')\big)$ is a strategy decomposition of V for Dis.

Suppose now that $R' = \emptyset$. The non-empty set R contains only vertices v such that $\varphi(v) = \infty$. Then, by definition and because all those vertices are associated with the same tuple, they must all have priority b' or $b' + 1$ for some even number b'.

Any edge going out from a vertex of R is progressive if and only if it fulfills condition 2. of the definition. Thus, the priority of all the vertices in R has to be even and is consequently b' with $b' < b$.

Let $R'' = \{u \in V : \mu(v) = \mu(u)|_{\pi(v)} \text{ for } v \in R\}$. (If we picture the tuples on a tree as in Figure 3, those are the vertices that are mapped to the nodes in the subtree rooted in the node corresponding to R.) By definition, the priority of all those vertices is also smaller than b. Moreover, an edge going out from a vertex in R'' can only be progressive if it goes to a vertex in R'' too. So, R'' is a trap for Con and an edge from a vertex in R'' owned by Dis to a vertex not in R'' cannot be progressive. So the progress measure on V gives also a progress measure on R'' when restricted to its vertices. By the inductive hypothesis, there is a strategy decomposition ω'' of R'' for Dis. Let T be the set of vertices from which Dis has a strategy to reach R'' and let τ be a corresponding positional reachability strategy. Let $U = V \setminus (R'' \cup T)$. Because, by construction, there is no edge in U from a vertex owned by Dis to a vertex in $R'' \cup T$, the progress measure on V gives also a progress measure on U when restricted to its vertices. By the inductive hypothesis, there is a strategy decomposition ω of U for Dis. Note that $\big((U, \omega), (T, \tau), (R'', \omega'')\big)$ is a strategy decomposition of V for Dis. □

3.3 From strategy decompositions to progress measures

Lemma 3.2. *If there is a strategy decomposition of V for Dis then there is a progress measure.*

Proof. The proof is by induction on the size of the game graph. Let b be the biggest vertex priority in V. We strengthen the inductive hypothesis by requiring that the progress measure (μ, φ) whose existence is claimed in the lemma is such that all sequences in the image of μ have the same prefix corresponding to indices k, such that $k > b$. We need to consider two cases based on the parity of b.

Suppose first that b is even. Let $\omega = \big((R, \omega'), (T, \tau), B\big)$ be a b-decomposition of V for Dis. Since $B \neq \emptyset$, by the inductive hypothesis there is a progress measure (μ', φ') on R. For every vertex $v \in T$, define its τ-distance to B to be the largest number of edges on a path starting at v, consistent with τ, and whose only vertex in B is the last one. Let k be the largest such τ-distance, and we define T_i, $1 \leq i \leq k$, to be the set of vertices in T whose τ-distance to B is i.

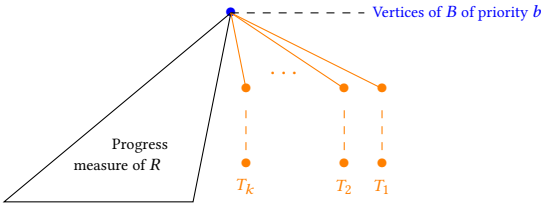


Figure 4. Construction of a progress measure - b even (the common prefix is not pictured).

Let $\langle m_{d-1}, m_{d-3}, \ldots, m_{b+1}\rangle$ be the common prefix of all sequences in the image of μ'. Let $t_1, t_2, \ldots, t_k$ be elements of the linearly ordered set used in progress measurements, such that for every r that is the component of a sequence in the image of μ' corresponding to priority $b-1$, we have $r > t_k > \cdots > t_2 > t_1$, and let t be a chosen element of the linearly ordered set (it does not matter which one). Define the progress labelling (μ, φ) for all vertices $v \in V$ as follows:

$$\big(\mu(v), \varphi(v)\big) = \begin{cases} \big(\mu'(v), \varphi'(v)\big) & \text{if } v \in R, \\ \big(\langle m_{d-1}, \ldots, m_{b+1}, t_i, m_{b-3}, \ldots, m_\ell\rangle, \infty\big) & \\ & \text{if } v \in T_i, 1 \leq i \leq k, \\ \big(\langle m_{d-1}, m_{d-3}, \ldots, m_{b+1}\rangle, \infty\big) & \text{if } v \in B; \end{cases}$$

where ℓ is the smallest odd number no smaller than $\pi(v)$ and $m_{b-3} = \ldots = m_\ell = t$.

The progress labelling (μ, φ) as defined above is a desired progress measure. It is illustrated as a tree in Figure 4.

Suppose now that b is odd. Let $\omega = \big((U, \omega''), (T, \tau), (R, \omega')\big)$ be a b-decomposition of V for Dis. Define τ-distances, sets T_i, and elements t_i and t for $1 \leq i \leq k$, in the analogous way to the "even b" case, replacing set B by set R. By the inductive hypothesis, there is a progress measure (μ'', φ'') on U, and let $\langle m_{d-1}, m_{d-3}, \ldots, m_{b+2}\rangle$ be the common prefix of all sequences in the image of μ''. We define a progress labelling (μ, φ) for all vertices in $U \cup T$ as follows:

$$\big(\mu(v), \varphi(v)\big) = \begin{cases} \big(\mu''(v), \varphi''(v)\big) & \text{if } v \in U, \\ \big(\langle m_{d-1}, \ldots, m_{b+2}, t_i, m_{b-3}, \ldots, m_\ell\rangle, \infty\big) & \\ & \text{if } v \in T_i, 1 \leq i \leq k; \end{cases}$$

where ℓ is the smallest odd number no smaller than $\pi(v)$ and $m_{b-3} = \ldots = m_\ell = t$.

If ω' is a b'-decomposition of R for $b' < b$ (case 2(b)i), then by the inductive hypothesis, there is a progress measure (μ', φ') on R. Without loss of generality, assume that all sequences in the images of μ' and of μ'' have the common prefix $\langle m_{d-1}, m_{d-3}, \ldots, m_{b+2}\rangle$, and that for all u and r that are the components of a sequence in the images of μ'' and μ', respectively, corresponding to priority b, we have $u > t_k > t_{k-1} > \cdots > t_1 > r$. Define the progress labelling (μ, φ) for all vertices $v \in R$ in the following way:

$$\big(\mu(v), \varphi(v)\big) = \big(\mu'(v), \varphi'(v)\big).$$

This is illustrated in Figure 5.

If, instead, ω' is a positional strategy for Dis that is mean-payoff winning for him on R (case 2(b)ii), then by the result of Brim et al. [2], there is an energy progress measure $\widehat{\varphi}$ for Dis on R. Let r' be such that $r' < t_1$, and define the progress labelling (μ, φ) for all

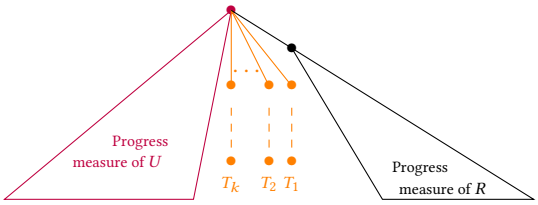


Figure 5. Construction of a progress measure - b odd - case 2(b)i.

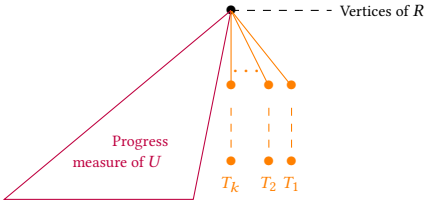


Figure 6. Construction of a progress measure - case 2(b)ii.

vertices $v \in R$ in the following way:

$$\big(\mu(v), \varphi(v)\big) = \big(\langle m_{d-1}, m_{d-3}, \ldots, m_{b+2}, r'\rangle, \widehat{\varphi}(v)\big).$$

This is illustrated in Figure 6.

The progress labelling (μ, φ) as defined above is a desired progress measure. □

4 Computing progress measures by lifting

In this section, we give a so-called lifting algorithm which identifies the winning sets for Dis and for Con by computing a progress measure on the winning set for Dis.

By the *tree* of a progress labelling (μ, φ), we mean the ordered tree whose nodes are all prefixes of all sequences $\mu(v)$ as v ranges over the vertices of the game graph, and such that every vertex v labels the node $\mu(v)$ of the tree. Let us say that progress labellings (μ, φ) and (μ', φ') are *isomorphic* if and only if their (partially labelled ordered) trees are isomorphic and $\varphi = \varphi'$.

We shall work with the following ordering on finite binary strings:

$$0s < \varepsilon, \quad \varepsilon < 1s, \quad bs < bs' \text{ if and only if } s < s',$$

where ε denotes the empty string, b ranges over binary digits, and s, s' range over binary strings.

Recall that n is the number of vertices, and d (assumed even) is the number of priorities.

Let $S_{n,d}$ be all sequences $\langle m_{d-1}, m_{d-3}, \ldots, m_\ell\rangle$ of binary strings such that:

- ℓ is odd and $1 \le \ell \le d+1$;
- $\sum_{i=\ell}^{d-1} |m_i| \le \lceil \lg n \rceil$;

and let us call a progress measurement, labelling or measure *succinct* if and only if all the sequences $\langle m_{d-1}, m_{d-3}, \ldots, m_\ell\rangle$ involved are members of $S_{n,d}$.

Lemma 4.1. *For every progress labelling, there exists a succinct isomorphic one.*

Proof. This is an immediate consequence of [19, Lemma 1], since for every progress labelling, its tree is of height at most $d/2$ and has at most n leaves. □

Corollary 4.2. *Lemmas 3.1 and 3.2 hold when restricted to succinct progress measures.*

We now order progress measurements lexicographically:

$$\big(\langle m_{d-1}, m_{d-3}, \ldots, m_\ell\rangle, e\big) < \big(\langle m'_{d-1}, m'_{d-3}, \ldots, m'_{\ell'}\rangle, e'\big)$$
$$\text{if and only if}$$
$$\text{either } \langle m_{d-1}, m_{d-3}, \ldots, m_\ell\rangle < \langle m'_{d-1}, m'_{d-3}, \ldots, m'_{\ell'}\rangle,$$
$$\text{or } \langle m_{d-1}, m_{d-3}, \ldots, m_\ell\rangle = \langle m'_{d-1}, m'_{d-3}, \ldots, m'_{\ell'}\rangle \text{ and } e < e'$$

and we extend them by a new greatest progress measurement $(\top, \infty)$. We then revise the set of progress labellings to allow the extended progress measurements, and we (partially) order it pointwise:

$$(\mu, \varphi) \le (\mu', \varphi') \text{ if and only if,}$$
$$\text{for all } v \in V, \big(\mu(v), \varphi(v)\big) \le \big(\mu'(v), \varphi(v')\big).$$

We also revise the definition of a progress measure by stipulating that an edge (v, u) which involves the progress measurement $(\top, \infty)$ is progressive if and only if the progress measurement of v is $(\top, \infty)$.

For any succinct progress labelling (μ, φ) and edge (v, u), we set $\mathrm{lift}(\mu, \varphi, v, u)$ to be the minimum succinct progress measurement $\big(\langle m_{d-1}, m_{d-3}, \ldots, m_\ell\rangle, e\big)$ which is at least $\big(\mu(v), \varphi(v)\big)$ and such that (v, u) is progressive in the updated succinct progress labelling

$$\Big(\mu\big[v \mapsto \langle m_{d-1}, m_{d-3}, \ldots, m_\ell\rangle\big], \varphi[v \mapsto e]\Big).$$

For any vertex v, we define an operator Lift_v on succinct progress labellings as follows:

$$\mathrm{Lift}_v(\mu, \varphi)(w) = \begin{cases} \big(\mu(w), \varphi(w)\big) & \text{if } w \neq v, \\ \min_{(v,u)\in E} \mathrm{lift}(\mu, \varphi, v, u) & \text{if Dis owns } w = v, \\ \max_{(v,u)\in E} \mathrm{lift}(\mu, \varphi, v, u) & \text{if Con owns } w = v. \end{cases}$$

Theorem 4.3 (Correctness of lifting algorithm)**.**

1. *The set of all succinct progress labellings ordered pointwise is a complete lattice.*
2. *Each operator* Lift_v *is inflationary and monotone.*
3. *From every succinct progress labelling* (μ, φ)*, every sequence of applications of operators* Lift_v *eventually reaches the least simultaneous fixed point of all* Lift_v *that is greater than or equal to* (μ, φ)*.*
4. *A succinct progress labelling* (μ, φ) *is a simultaneous fixed point of all operators* Lift_v *if and only if it is a succinct progress measure.*
5. *If* (μ^*, φ^*) *is the least succinct progress measure, then* $\{v : \big(\mu^*(v), \varphi^*(v)\big) \neq (\top, \infty)\}$ *is the set of winning positions for Dis.*

Proof.

1. The partial order of all succinct progress labellings is the pointwise product of n copies of the finite linear order of all succinct progress measurements.
2. We have inflation, i.e. $\mathrm{Lift}_v(\mu, \varphi)(w) \ge \big(\mu(w), \varphi(w)\big)$, by the definitions of $\mathrm{Lift}_v(\mu, \varphi)(w)$ and $\mathrm{lift}(\mu, \varphi, v, u)$.
   For monotonicity, supposing $(\mu, \varphi) \le (\mu', \varphi')$, it suffices to show that, for every edge (v, u), we have $\mathrm{lift}(\mu, \varphi, v, u) \le$

1. Initialise (μ, φ) to the least succinct progress labelling $(v \mapsto \langle\rangle, v \mapsto 0)$
2. While $\mathrm{Lift}_v(\mu, \varphi) \neq (\mu, \varphi)$ for some v, update (μ, φ) to become $\mathrm{Lift}_v(\mu, \varphi)$.
3. Return the set $W_{\mathrm{Dis}} = \{v : \big(\mu(v), \varphi(v)\big) \neq (\top, \infty)\}$ of winning positions for Dis.

Table 1. The lifting algorithm.

$\mathrm{lift}(\mu', \varphi', v, u)$, which is in turn implied by the straightforward observation that, whenever an edge is progressive with respect to a progress labelling, it remains progressive after any lessening of the progress measurement of its target vertex.

3. This holds for any family of inflationary monotone operators on a finite complete lattice. Consider any such maximal sequence from (μ, φ). It is an upward chain from (μ, φ) to some (μ^*, φ^*) which is a simultaneous fixed point of all the operators. For any $(\mu', \varphi') \geq (\mu, \varphi)$ which is also a simultaneous fixed point, a simple induction confirms that $(\mu^*, \varphi^*) \leq (\mu', \varphi')$.
4. Here we have a rewording of the definition of a succinct progress measure.
5. Let $W = \{v : \big(\mu^*(v), \varphi^*(v)\big) \neq (\top, \infty)\}$. The set of winning positions for Dis is contained in W by Lemma 2.3, Lemma 3.2 and Corollary 4.2, because (μ^*, φ^*) is the least succinct progress measure.
   Since (μ^*, φ^*) is a progress measure, we have that, for every progressive edge (v, u), if $\big(\mu^*(v), \varphi^*(v)\big) \neq (\top, \infty)$ then $\big(\mu^*(u), \varphi^*(u)\big) \neq (\top, \infty)$. In order to show that Dis has a winning strategy from every vertex in W, it remains to apply Lemmas 3.1 and 2.1 to the subgame consisting of the vertices in W.

□

Lemma 4.4 (Jurdziński and Lazić [19])**.** *Depending on the asymptotic growth of d as a function of n, the size of the set $S_{n,d}$ is as follows:*

1. $O\left(n^{1+o(1)}\right)$ *if* $d = o(\log n)$;
2. $\Theta\left(n^{\lg(\delta+1)+\lg(e_\delta)+1} \Big/ \sqrt{\log n}\right)$ *if* $d/2 = \lceil \delta \lg n \rceil$, *for some positive constant* δ, *and where* $e_\delta = (1 + 1/\delta)^\delta$;
3. $O\left(dn^{\lg(d/\lg n)+\lg e+o(1)}\right)$ *if* $d = \omega(\log n)$.

Theorem 4.5 (Complexity of lifting algorithm)**.** *Depending on the asymptotic growth of d as a function of n, the running time of the algorithm is as follows:*

1. $O\left(mn^{2+o(1)}C\right)$ *if* $d = o(\log n)$;
2. $O\left(mn^{\lg(\delta+1)+\lg(e_\delta)+2}C \cdot \log d \cdot \sqrt{\log n}\right)$ *if* $d \leq 2\lceil \delta \lg n \rceil$, *for some positive constant* δ;
3. $O\left(dmn^{\lg(d/\lg n)+2.45}C\right)$ *if* $d = \omega(\log \eta)$.

The algorithm works in space $O(n \cdot \log n \cdot \log d)$.

Proof. The work space requirement is dominated by the number of bits needed to store a single succinct progress labelling, which is at most $n(\lceil \lg n \rceil \lceil \lg d \rceil + \lceil \lg(nC) \rceil)$.

Since bounded-depth successors of elements of $S_{n,d}$ are computable in time $O(\log n \cdot \log d)$ (cf. the proof of [19, Theorem 7], the Lift_v operators can be implemented to work in time $O(\deg(v) \cdot (\log n \cdot \log d + \log C))$. It then follows, observing that the algorithm lifts each vertex at most $|S_{n,d}|(nC + 1)$ times, that its running time is bounded by

$$O\left(\sum_{v \in V} \deg(v) \cdot (\log n \cdot \log d + \log C)|S_{n,d}|(nC + 1)\right) = O\left(mnC(\log n \cdot \log d + \log C)|S_{n,d}|\right) .$$

From there, the various stated bounds are obtained by applying Lemma 4.4, and by suppressing some of the multiplicative factors that are logarithmic in the bit-size of the input. Suppressing the $\log C$ factor is justified by using the unit-cost RAM model, which is the industry standard in algorithm analysis. The reasons for suppressing the $\log n$ and $\log d$ factors are more varied: in case 1, they are absorbed by the $o(1)$ term in the exponent of n, and in case 3, they are absorbed in the 2.45 term in the exponent of n, because $\lg e < 1.4427$. □

5 From winning sets to strategy decompositions for Con

The pseudo-quasi-polynomial lifting algorithm computes the least progress measure and hence, by Lemmas 3.1 and 2.1, it can be easily adapted to synthesize a winning strategy for Dis from all vertices in her winning set. In this section we tackle the problem of strategy synthesis for Con. By (the proof of) Lemma 2.2, in order to synthesize a winning strategy for Con, it suffices to compute a strategy decomposition for him. We argue that this can also be achieved in pseudo-quasi-polynomial time.

Theorem 5.1 (Complexity of computing strategy decompositions)**.** *There is a pseudo-quasi-polynomial algorithm that computes strategy decompositions for both players on their winning sets.*

In order to establish that strategy decompositions for Con can be computed in pseudo-quasi-polynomial time, it suffices to prove the following lemma, because the polynomial-time oracle algorithm becomes a pseudo-quasi-polynomial algorithm, once the oracle for computing winning strategies in mean-payoff games is replaced by a pseudo-polynomial algorithm [2, 8, 25], and the oracle for computing the winning sets in mean-payoff parity games is replaced by the pseudo-quasi-polynomial procedure from Section 4.

Lemma 5.2. *There is a polynomial-time algorithm, with oracles for computing winning strategies in mean-payoff games and for computing winning sets in mean-payoff parity games, that computes a strategy decomposition for Con of his winning set.*

Proof. Without loss of generality, we may assume that Con has a winning strategy from every vertex in V, since a single call to the oracle allows us to reduce V to the subgame corresponding to the winning set for Con.

Below, we describe a recursive procedure for computing a strategy decomposition for Con of the set of all vertices, that has a similar structure to the inductive proof of Lemma 2.3. In parallel

with the description of the recursive procedure, we elaborate an inductive proof that it does indeed compute a strategy decomposition for Con on V.

Note that our procedure avoids incurring the penalty of adding to its running time a factor that is exponential in the number of distinct vertex priorities, by repeatedly using the oracle for computing the winning sets in appropriately chosen subgames. We give a detailed analysis of the worst-case running time at the end of this proof.

Let B be the set of vertices of the highest priority b; let T be the set of vertices (not including vertices in B) from which Dis has a strategy to reach a vertex in B; let τ be a corresponding positional rechability strategy; and let $R = V \setminus (B \cup T)$. We consider two cases, depending on the parity of b.

Even b**.** Call the oracle to obtain the partition R_{Con} and R_{Dis} of R, the winning sets for Con and for Dis, respectively, in the subgame R. We argue that $R_{\mathrm{Con}} \neq \emptyset$. Otherwise, by Lemma 2.3, there is a strategy decomposition ω of R for Dis, and hence $\big((R,\omega),(T,\tau),B\big)$ is a strategy decomposition of V for Dis, which, by Lemma 2.1, contradicts the assumption that Con has a winning strategy from every vertex.

Let T' be the set of vertices (not including vertices in R_{Con}) from which Con has a strategy to reach a vertex in R_{Con}, and let τ' be a corresponding positional reachability strategy, and let $U = V \setminus (R_{\mathrm{Con}} \cup T')$. By the inductive hypothesis, a recursive call of our procedure on R_{Con} will produce a strategy decomposition ω' of R_{Con} for Con, and another recursive call of the procedure on U will produce a strategy decomposition ω'' of U for Con. We claim that $\big((U,\omega''),(T',\tau'),(R_{\mathrm{Con}},\omega')\big)$ is a strategy decomposition of V for Con.

Odd b**.** Call the oracle for computing positional winning strategies in mean-payoff games to obtain a positional strategy λ for Con that is mean-payoff winning for him on V; such a strategy exists because Con has a mean-payoff parity winning strategy from every vertex. Since R is a trap for Con, it must be the case that Con has a winning strategy from every vertex in the subgame R. By the inductive hypothesis, a recursive call of our procedure on R will produce a strategy decomposition ω' of R for Con. We claim that $\big((R,\omega'),(T,\tau),B,\lambda\big)$ is a strategy decomposition of V for Con.

It remains to argue that the recursive procedure described above works in polynomial time in the worst case. Observe that in both cases considered above, a call of the procedure on a game results in two or one recursive calls, respectively. In both cases, the recursive calls are applied to subgames with strictly fewer vertices, and—crucially for the complexity analysis—in the former case, the two recursive calls are applied to subgames on disjoint sets of vertices. Additional work (other than recursive calls and oracle calls) in both cases can be bounded by $O(m)$, since the time needed is dominated by the worst case bound on the computation of reachability strategies. Overall, the running time function $T(n)$ of the recursive procedure, where n is the number of vertices in the input game graph, satisfies the following recurrence:

$$T(n) \leq T(n') + T(n'') + O(m), \qquad \text{where } n' + n'' < n,$$

and hence $T(n) = O(nm)$. □

6 Conclusion

Our main result is the first pseudo-quasi-polynomial algorithm for computing the values of mean-payoff parity games, and hence also for deciding the winner in energy parity games and in parity games with weights. The main technical tools that we introduce to achieve the main result are strategy decompositions and progress measures for the threshold version of mean-payoff games. We believe that our techniques can be adapted to also produce optimal strategies for both players (i.e., the strategies that secure the value that we show how to compute). Another direction for future work is improving the complexity of solving stochastic mean-payoff parity games [5].

Acknowledgements

This research has been supported by the EPSRC grant EP/P020992/1 (Solving Parity Games in Theory and Practice).

References

[1] P. Bouyer, N. Markey, J. Olschewski, and M. Ummels. 2011. Measuring permissiveness in parity games: Mean-payoff parity games revisited. In *ATVA*. 135–149.
[2] L. Brim, J. Chaloupka, L. Doyen, R. Gentilini, and J.-F. Raskin. 2011. Faster algorithms for mean-payoff games. *Form. Methods Syst. Des.* 38, 2 (2011), 97–118.
[3] C. S. Calude, S. Jain, B. Khoussainov, W. Li, and F. Stephan. 2017. Deciding parity games in quasipolynomial time. In *STOC*. 252–263.
[4] K. Chatterjee and L. Doyen. 2012. Energy parity games. *Theoretical Computer Science* 458 (2012), 49–60.
[5] K. Chatterjee, L. Doyen, H. Gimbert, and Y. Oualhadj. 2014. Perfect-information stochastic mean-payoff parity games. In *FOSSACS*. 210–225.
[6] K. Chatterjee, M. Henzinger, and A. Svozil. 2017. Faster algorithms for mean-payoff parity games. In *MFCS*. 39:1–39:17.
[7] K. Chatterjee, T. A. Henzinger, and M. Jurdziński. 2005. Mean-payoff parity games. In *LICS*. 178–187.
[8] C. Comin and R. Rizzi. 2017. Improved pseudo-polynomial bound for the value problem and optimal strategy synthesis in mean payoff games. *Algorithmica* 77, 4 (2017), 995–1021.
[9] A. Condon. 1992. The complexity of stochastic games. *Information and Computation* 96, 2 (1992), 203–224.
[10] L. Daviaud, M. Jurdziński, and M. Lazić. 2018. A pseudo-quasi-polynomial algorithm for solving mean-payoff parity games. arXiv:1803.04756. (2018).
[11] A. Ehrenfeucht and J. Mycielski. 1979. Positional strategies for mean payoff games. *Journal of Game Theory* 8, 2 (1979), 109–113.
[12] E. A. Emerson and C. Jutla. 1991. Tree automata, mu-calculus and determinacy. In *FOCS*. 368–377.
[13] E. A. Emerson, C. Jutla, and A. P. Sistla. 2001. On model-checking for fragments of μ-calculus. *Theoretical Computer Science* 258, 1–2 (2001), 491–522.
[14] J. Fearnley, S. Jain, S. Schewe, F. Stephan, and D. Wojtczak. 2017. An ordered approach to solving parity games in quasi polynomial time and quasi linear space. In *SPIN*. 112–121.
[15] N. Fijalkow and M. Zimmermann. 2014. Parity and Streett games with costs. *Logical Methods in Computer Science* 10, 1:14 (2014), 1–29.
[16] Y. Gurevich and L. Harrington. 1982. Trees, automata, and games. In *STOC*. 60–65.
[17] D. S. Johnson. 2007. The NP-completeness column: Finding needles in haystacks. *ACM Transactions on Algorithms* 3, 2 (2007).
[18] M. Jurdziński. 2000. Small progress measures for solving parity games. In *STACS*. 290–301.
[19] M. Jurdziński and R. Lazić. 2017. Succinct progress measures for solving parity games. In *LICS*. 1–9.
[20] N. Klarlund and D. Kozen. 1995. Rabin measures. *Chicago Journal of Theoretical Computer Science* (1995). Article 3.
[21] R. McNaughton. 1993. Infinite games played on finite graphs. *Annals of Pure and Applied Logic* 65, 2 (1993), 149–184.
[22] S. Schewe, A. Weinert, and M. Ziemmermann. 2018. Parity games with weights. arXiv:1804.06168. (2018).
[23] W. Thomas. 1995. On the synthesis of strategies in infinite games. In *STACS*. 1–13.
[24] W. Zielonka. 1998. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theoretical Computer Science* 200 (1998), 135–183.
[25] U. Zwick and M. Paterson. 1996. The complexity of mean-payoff games on graphs. *Theoretical Computer Science* 158 (1996), 343–359.