Extending finite memory determinacy

General techniques and an application to energy parity games

Stéphane Le Roux   Arno Pauly *

Université Libre de Bruxelles, Brussels, Belgium

Stephane.Le.Roux@ulb.ac.be   Arno.Pauly@cl.cam.ac.uk

Abstract

We provide several techniques to extend finite memory determinacy from some restricted class of games played on a finite graph to a larger class. As a particular example, we study energy parity games.

First we show that under some general conditions the finite memory determinacy of a class of two-player win/lose games played on finite graphs implies the existence of a Nash equilibrium built from finite memory strategies for the corresponding class of multi-player multi-outcome games. This generalizes a previous result by Brihaye, De Pril and Schewe.

Then we investigate adding additional constraints to the winning conditions, in a way that generalizes the move from parity to bounded energy parity games. We prove that under some conditions, this preserves finite memory determinacy. We show that bounded energy parity games and unbounded energy parity games are equivalent, and thus obtain a new proof of finite memory determinacy for energy parity games. Our proof yields significantly improved bounds on the memory required compared to the original one by Chatterjee and Doyen. We then apply our main theorem to show that multi-player multi-outcome energy parity games have finite memory Nash equilibria.

Our proofs are generally constructive, that is, provide upper bounds for the memory required, as well as algorithms to compute the relevant winning strategies.

1. Introduction

The usual model employed for synthesis are sequential two-player win/lose games played on finite graphs. The vertices of the graph correspond to states of a system, and the two players jointly generate an infinite path through the graph (the *run*). One player, the protagonist, models the aspects of the system under the control of the designer. In particular, the protagonist will win the game iff the run satisfies the intended specification. The other player is assumed to be fully antagonistic, thus wins iff the protagonist loses. One then would like to find winning strategies of the protagonist, that is, a strategy for her to play the game in such a way that she will win regardless of the antagonist's moves. Particularly desirable winning strategies are those which can be executed by a finite automaton.

Classes of games are distinguished by the way the winning conditions (or more generally, preferences of the players) are specified. Typical examples include:

- Muller conditions, where only the set of vertices visited infinitely many times matters;
- Parity conditions, where each vertex has a priority, and the winner is decided by the parity of the least priority visited infinitely many times;
- Energy conditions, where each vertex has an energy delta (positive or negative), and the protagonist loses if the cumulative energy values ever drop below 0;
- Discounted payoff conditions, where each vertex has a payoff value, and the outcome is determined by the discounted sum of all payoffs visited with some discount factor $0 < \lambda < 1$;
- Combinations of these, such as energy parity games, where the protagonist has to simultaneously ensure that the least parity visited infinitely many times is odd and that the cumulative energy value is never negative.
- Cost Streett conditions (Fijalkow and Zimmermann 2012, 2014), where every odd label needs to be followed by an even label within a given distance for player 1 to win.

Our first goal is to dispose of two restrictions of this setting: First, we would like to consider any number of players; and second allow them to have far more complicated preferences than just preferring winning over losing. The former generalization is crucial in a distributed setting (also e.g. (Brihaye et al. 2013; Bulling and Goranko 2013)): If different designers control different parts of the system, they may have different specifications they would like to enforce, which may be partially but not entirely overlapping. The latter seems desirable in a broad range of contexts. Indeed, rarely is the intention for the behaviour of a system formulated entirely in black and white: We prefer a programm just crashing to accidently erasing our hard-drive; we prefer a programm to complete its task in 1 minute to it taking 5 minutes, etc.

Rather than achieving this goal by revisiting each individual type of game and proving the desired results directly (e.g. by generalizing the original proofs of the existence of winning strategies), we shall provide a transfer theorem: In Theorem 5, we will show that (under some conditions), if the two-player win/lose version of a game is finite memory determined, the corresponding multi-player multi-outcome games all have finite memory Nash equilibria.

This result is more general than a similar one obtained by BRIHAYE, DE PRIL and SCHEWE (Brihaye et al. 2013),(Pril 2013, Theorem 4.4.14). A particular class of games covered by our result but not the previous one are energy parity games as introduced

* The authors were supported by the ERC inVest project.

by CHATTERJEE and DOYEN (Chatterjee and Doyen 2012). The high-level proof idea follows earlier work by the authors on equilibria in infinite sequential games, using Borel determinacy as a blackbox (Le Roux and Pauly 2014a,b)[1] – unlike the constructions there (cf. (Le Roux and Pauly 2015, 2014c)), the present ones however are constructive and thus give rise to algorithms computing the equilibria in the multi-player multi-outcome games given suitable winning strategies in the two-player win/lose versions.

Echoing DE PRIL in (Pril 2013), we would like to stress that our conditions apply to the preferences of each player individually. For example, some players could pursue energy parity conditions, whereas others have preferences based on Muller conditions: Our results apply just as they would do if all players had preferences of the same type.

Our second main line of investigation (in Section 6) is about putting additional constraints on one player in a two-player win/lose game. We provide a number of results showing that if the original games had finite-memory winning strategies, then additionally requiring a player to either meet or avoid the histories accepted by some finite automaton yields a new class of games that is still finite-memory determined. This construction is a generalization of the addition of *bounded* energy constraints – our Example 38 shows that the corresponding result for adding unbounded energy constraints is actually false.

For the particular case of parity or Muller games as the starting point, we prove in Section 4 that the bounded energy and unbounded energy versions are equivalent. Using similar arguments as those in Section 6, we then give a new proof of finite-memory determinacy of energy parity games as Theorem 30. Our proof shows that $\log 2nW$ of memory suffice, in contrast, (Chatterjee and Doyen 2012, Theorem 1) requires $\log 4ndW$. Here n is the number of vertices, W the largest energy delta, and d the number of priorities. In particular, the required memory does not increase in size with the complexity of the parity winning condition. Already (Chatterjee and Doyen 2012, Lemma 1) establishes an $\log nW$ lower bound for the memory, thus, up to $\mathcal{O}$-equivalence, our construction is optimal.

A further benefit of our new proof in Theorem 30 is that it yields a more general result applying also to disjunctions of energy parity winning conditions. This allows us to then apply the Theorem 5 and to conclude that multi-player multi-outcome bounded energy parity games have Nash equilibria built from finite-memory strategies.

2. Background

A two-player win/lose game played on a finite graph is specified by a directed graph (V, E) where every vertex has an outgoing edge, a starting vertex $v_0 \in V$, two sets $V_1 \subseteq V$ and $V_2 := V \setminus V_1$, and a *winning condition* $W \subseteq V^\omega$. Starting from v_0, the players move a token along the graph, ω times, with player $a \in \{1, 2\}$ picking and following an outgoing edge whenever the current vertex lies in V_a. Player 1 wins iff the infinite sequence of visited vertices is in W.

For $a \in \{1, 2\}$ let $\mathcal{H}_a$ be the set of finite paths in (V, E) starting at v_0 and ending in V_a. Let $\mathcal{H} := \mathcal{H}_1 \cup \mathcal{H}_2$ be the possible *finite histories* of the game, and let $[\mathcal{H}]$ be the infinite ones. For clarity we may write $[\mathcal{H}_g]$ instead of $[\mathcal{H}]$ for a game g. A *strategy* of player $a \in \{1, 2\}$ is a function of type $\mathcal{H}_a \to V$ such that $(v, s(hv)) \in E$ for all $hv \in \mathcal{H}_a$. A pair of strategies (s_1, s_2) for the two players induces a run $\rho \in V^\omega$: Let $s := s_1 \cup s_2$ and set $\rho(0) := v_0$ and $\rho(n+1) := s(\rho(0)\rho(1)\dots\rho(n))$. For all strategies s_a of player a let $\mathcal{H}(s_a)$ be the finite histories in $\mathcal{H}$ that are compatible with s_a, and let $[\mathcal{H}(s_a)]$ be the infinite ones. A strategy s_a is said to be winning if $[\mathcal{H}(s_a)] \subseteq W$, *i.e.* a wins regardless of her opponent's moves.

A *strategic implementation* for player a using m bits of memory is a function $\sigma : V \times \{0,1\}^m \to V \times \{0,1\}^m$ that describes the two simultaneous updates of player a upon arrival at a vertex v if its memory content was M just before arrival: $(v, M) \mapsto \pi_2 \circ \sigma(v, M)$ describes the memory update and $(v, M) \mapsto \pi_1 \circ \sigma(v, M)$ the choice for the next vertex. This choice will be ultimately relevant only if $v \in V_a$, in which case we require that $(v, \pi_1 \circ \sigma(v, M)) \in E$.

Together with an initial memory content $M_\epsilon \in \{0,1\}^m$, a strategic implementation provides a finite memory strategy. The memory content after some history is defined by induction: $M_\sigma(M_\epsilon, \epsilon) := M_\epsilon$ and $M_\sigma(M_\epsilon, hv) := \pi_2 \circ \sigma(v, M_\sigma(M_\epsilon, h))$ for all $hv \in \mathcal{H}$. The *finite-memory strategy* s_a induced by the strategic implementation σ together with initial memory content M_ϵ is defined by $s_a(hv) := \pi_1 \circ \sigma(v, M_\sigma(M_\epsilon, h))$ for all $hv \in \mathcal{H}_a$. If not stated otherwise, we will assume the initial memory to be 0^m.

A (general) game played on a finite graph is specified by a directed graph (V, E), a set of agents A, a cover $\{V_a\}_{a \in A}$ of V via pairwise disjoint sets, the starting vertex v_0, and for each player a a preference relation $\prec_a \subseteq [\mathcal{H}] \times [\mathcal{H}]$. The notions of strategies and induced runs generalize in the obvious way. In particular, instead of a pair of strategies (one per player), we consider families $(s_a)_{a \in A}$, which are called strategy profiles.

The concept of a winning strategy no longer applies though. Instead, we use the more general notion of a Nash equilibrium: A family of strategies $(s_a)_{a \in A}$ is a Nash equilibrium, if there is no player $a_0 \in A$ and alternate strategy s'_{a_0} such that a would prefer the run induced by $(s_a)_{a \in A \setminus \{a_0\}} \cup (s'_a)_{a \in \{a_0\}}$ to the run induced by $(s_a)_{a \in A}$. Intuitively, no player can gain by unilaterally deviating from a Nash equilibrium. Note that the Nash equilibria in two-player win/lose games are precisely those pairs of strategy where one strategy is a winning strategy.

The transfer of results from the two-player win/lose case to the general case relies on the idea that each general game induces a collection of two-player win/lose games, namely the threshold games of the future games, as below.

Definition 1 (Future game and one-vs-all threshold game)**.** Let $g = \langle (V, E), v_0, A, \{V_a\}_{a \in A}, (\prec_a)_{a \in A} \rangle$ be a game played on a finite graph.

- Let $a_0 \in A$ and $\rho \in [\mathcal{H}]$, the one-vs-all threshold game $g_{a_0,\rho}$ for a_0 and ρ is the win-lose two-player game played on (V, E), starting at v_0, with vertex subsets V_{a_0} and $\bigcup_{a \in A \setminus \{a_0\}} V_a$, and with winning set $\{\rho' \in [\mathcal{H}] \mid \rho \prec_{a_0} \rho'\}$ for Player 1.
- Let $v \in V$. For paths hv and vh' in (V, E) let $hv\hat{\ }vh' := hvh'$.
- For all $h \in \mathcal{H}$ with last vertex v let $g^h := \langle (V, E), v, A, \{V_a\}_{a \in A}, (\prec_a^h)_{a \in A} \rangle$ be called the future game of g after h, where for all $\rho, \rho' \in [\mathcal{H}_{g^h}]$ we set $\rho \prec_a^h \rho'$ iff $h\hat{\ }\rho \prec_a h\hat{\ }\rho'$. If s is a strategy in g, let s^h be the strategy in g^h such that $s^h(h') := s(h\hat{\ }h')$ for all $h' \in \mathcal{H}_{g^h}$.

Observation 2. Let $g = \langle (V, E), v_0, A, \{V_a\}_{a \in A}, (\prec_a)_{a \in A} \rangle$ be a game played on a finite graph.

1. g and its thresholds games have the same strategies.
2. for all $h, h' \in \mathcal{H}$ ending with the same vertex the games g^h and $g^{h'}$ have the same (finite-memory) strategies.
3. g, its future games, and their thresholds games have the same strategic implementations.
4. If a strategy s_a in g is finite-memory, for all $h \in \mathcal{H}$ the strategy s_a^h is also finite-memory.

[1] Precursor ideas are also present in (Le Roux 2013) and (Mertens 1987) (the specific result in the latter was joint work with Neymann).

Proof. We only prove the fourth claim. Since s_a is a finite-memory strategy, it comes from some strategic implementation σ with initial memory M_ϵ. We argue that σ with initial memory $M_\sigma(M_\epsilon, h))$ implements s_a^{hv}: First, $s_a^{hv}(v) = s_a(hv) = \pi_1 \circ \sigma(v, M_\sigma(M_\epsilon, h)) = \pi_1 \circ \sigma(v, M_\sigma(M_\sigma(M_\epsilon, h), \epsilon))$; second, for all $h'v' \in \mathcal{H}^{hv}$ we have $s_a^{hv}(vh'v') = s_a(hvh'v') = \pi_1 \circ \sigma(v', M_\sigma(M_\epsilon, hvh')) = \pi_1 \circ \sigma(v', M_\sigma(M_\sigma(M_\epsilon, h), vh'))$. □

We will employ some additional restrictions on preferences: a preference relation $\prec \subseteq [\mathcal{H}] \times [\mathcal{H}]$ is called *prefix-linear*, if $\rho \prec \rho' \Leftrightarrow h\hat{}\rho \prec h\hat{}\rho'$ for all $\rho, \rho', h\hat{}\rho \in [\mathcal{H}]$. It is *prefix-independent*, if $\rho \prec \rho' \Leftrightarrow h\hat{}\rho \prec \rho'$ and $\rho' \prec \rho \Leftrightarrow \rho' \prec h\hat{}\rho$ for all $\rho, \rho', h\hat{}\rho \in [\mathcal{H}]$. Clearly, a prefix-independent preference is prefix-linear.

As a further generalization, we will consider *automatic-piecewise prefix-linear* preferences $\prec$. Here, there is an equivalence relation on $\mathcal{H}$ with equivalence classes (pieces for short) in $\overline{\mathcal{H}}$ and satisfying three constraints: First, the histories in the same piece end with the same vertex. Second, there exists a deterministic finite automaton, without accepting states, that reads histories and such that two histories are equivalent iff reading them leads to the same states. Third, for all $h\hat{}\rho, h\hat{}\rho', h'\hat{}\rho, h'\hat{}\rho' \in [\mathcal{H}]$, if $\overline{h'} = \overline{h} \in \overline{\mathcal{H}}$, then $h\hat{}\rho \prec h\hat{}\rho' \Leftrightarrow h'\hat{}\rho \prec h'\hat{}\rho'$.

Definition 3. A preference relation $\prec$ is automatic-piecewise Mont if there is an equivalence relation on $\mathcal{H}$ that is decidable by a finite automaton (with the first two constraints as for automatic-piecewise prefix linearity) such that the following holds. For every run $h_0\hat{}\rho \in [\mathcal{H}]$ that is regular (as a singleton language) and for every family $(h_n)_{n\in\mathbb{N}}$ of paths in (V, E) such that $h_0\hat{}h_1\hat{}\dots\hat{}h_n \in \overline{h_0}$ for all n, and such that $h_0\hat{}\dots\hat{}h_n\hat{}\rho \in [\mathcal{H}]$ for all n, if $h_0\hat{}\dots\hat{}h_n\hat{}\rho \prec h_0\hat{}\dots\hat{}h_{n+1}\hat{}\rho$ for all n then $h_0\hat{}\rho \prec h_0\hat{}h_1\hat{}h_2\hat{}h_3 \dots$.

For a cycle h starting and ending in some $v \in V$, let $h^{[0]} = v$ and $h^{[n+1]} = h^{[n]}\hat{}h$, and finally $h^{[\omega]} = \lim_{n\to\infty} h^{[n]}$. We call $\prec$ automatic-piecewise regular-Mont, if for any regular $h_0\hat{}\rho \in [\mathcal{H}]$ and cycle h in (V, E), if $\forall n \in \mathbb{N} \quad h_0\hat{}h^{[n]} \in \overline{h_0}$ and $\forall n \in \mathbb{N} \quad h_0\hat{}h^{[n]}\hat{}\rho \prec h_0\hat{}h^{[n+1]}\hat{}\rho$, then $h_0\hat{}\rho \prec h_0\hat{}h^{[\omega]}$.

We will also require preference relations to be strict weak orders, so we recall the corresponding definition:

Definition 4 (Strict weak order)**.** A relation $\prec$ is called a *strict weak order* if it satisfies:

$$\begin{array}{ll} \forall x, & \neg(x \prec x) \\ \forall x, y, z, & x \prec y \wedge y \prec z \Rightarrow x \prec z \\ \forall x, y, z, & \neg(x \prec y) \wedge \neg(y \prec z) \Rightarrow \neg(x \prec z) \end{array}$$

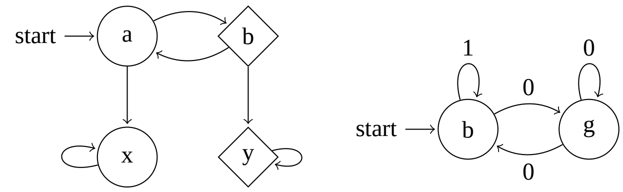
Strict weak orders capture in particular the situation where each player cares only about a particular aspect of the run (e.g. her associated personal payoff), and is indifferent between runs that coincide in this aspect but not others (e.g. the runs with identical associated payoffs for her, but different payoffs for the other players).

3. The transfer theorem

We will start this section with the statement of our first main result, showing how to transfer finite-memory determinacy from class of two-player win/lose games to the corresponding multi-player multi-outcome version. We informally sketch the proof. This is followed by the technical definitions and lemmata used in the formal proof, and then the proof itself. The section is completed by a discussion of the relevance and a comparison to prior results.

Theorem 5. Consider a game played by a set of players A on a finite graph such that

1. The $\prec_a$ are automatic-piecewise prefix-linear regular-Mont strict weak orders with k pieces.



(a) (b)

Figure 1

2. All one-vs-all threshold games of all future games are determined via strategies using m bits of memory.

Then the game has a Nash equilibrium in finite-memory strategies requiring $|A|(m + 2\log k) + 1$ bits of memory.

Definition 6 below rephrases Definitions 2.3 and 2.5 from (Le Roux 2013): The guarantee of a player is the smallest set of runs that is upper-closed w.r.t. the strict-weak-order preference of the player and includes every incomparability class (of the preference) that contains any run compatible with a given strategy of the player in the subgame at any given finite history of the game. The best guarantee of a player consists of the intersection of all her guarantees over the set of strategies.

More plainly spoken, the best guarantee for a player at some history is the set of runs such that the player cannot unilaterally enforce something better (for him). We will then show that each player has indeed a strategy enforcing her guarantee. Note that the notion of best guarantee for a player does not at all depend on the preferences of the other players; and as such, it is rather strenuous to consider such runs to be optimal in some sense (cf. Example 7). However, we can construct a Nash equilibrium by starting with a strategy profile where everyone is realizing their guarantee, and then adding punishments against any deviators.

Definition 6 (Player (best) future guarantee)**.** Let g be the game $\langle (V, E), v_0, A, \{V_a\}_{a\in A}, (\prec_a)_{a\in A}\rangle$ where $\prec_a$ is a strict weak order for some $a \in A$. For all $h \in \mathcal{H}$ and strategies s_a for a in g^h let $\gamma_a(h, s_a) := \{\rho \in [\mathcal{H}_{g^h}] \mid \exists \rho' \in [\mathcal{H}_{g^h}(s_a)], \neg(\rho \prec_a^h \rho')\}$ be the player future guarantee by s_a in g^h. Let $\Gamma_a(h) := \bigcap_{s_a} \gamma_a(h, s_a)$ be the best future guarantee of a in g^h.

Example 7. Let the underlying graph be as in Figure 1a, where circle vertices are controlled by Player 1 and diamond vertices are controlled by Player 2. The preference relation of Player 1 is $(ab)^\omega \succ_1 a(ba)^n x^\omega \succ_1 (ab)^n y^\omega$ and the preference relation of Player 2 is $(ab)^\omega \succ_2 (ab)^n y^\omega \succ_2 a(ba)^n x^\omega$ (in particular, both players care only about the tail of the run).

Then $\Gamma_1(a) = \{(ab)^\omega\} \cup \{a(ba)^n x^\omega \mid n \in \mathbb{N}\}$ and $\Gamma_2(a) = [\mathcal{H}]$. Player 1 realizing her guarantee means for her to move to x immediately, thus forgoing any chance of realizing the run $(ab)^\omega$. The Nash equilibrium constructed in the proof of Theorem 5 will be Player 1 moving to x and Player 2 moving to y. Note that in this particular game, the preferences of Player 2 have no impact at all on the Nash equilibrium that will be constructed.

Example 8. Lemma 9 below collects simple properties of the guarantee. The only assumption about the preference therein is the one already involved in Definition 6.

Lemma 9. Let g be a game on a graph, let $\prec_a$ be a strict weak order preference for some player a, let $h \in \mathcal{H}$, let s_a be a strategy for a in g^h, let $h' \in \mathcal{H}(s_a)$, and let s'_a be a strategy for a in $g^{h\hat{}h'}$.

1. Then $h'\hat{}\gamma_a(h\hat{}h', s_a^{h'}) \subseteq \gamma_a(h, s_a)$ for all $h' \in \mathcal{H}(s_a)$.

2. If $\gamma_a(h\hat{}h', s'_a) \subsetneq \gamma_a(h\hat{}h', s_a^{h'})$, there exists $\rho \in \gamma_a(h\hat{}h', s_a^{h'})$ such that $\rho \prec_a^{h\hat{}h'} \rho'$ for all $\rho' \in \gamma_a(h\hat{}h', s'_a)$.
3. If $\gamma_a(h\hat{}h', s'_a) \subseteq \gamma_a(h\hat{}h', s_a^{h'})$ then $h'\hat{}\gamma_a(h\hat{}h', s'_a) \subseteq \gamma_a(h, s_a)$.

Proof. 1. Let $\rho \in \gamma_a(h\hat{}h', s_a^{h'})$, so by Definition 6 there exists $\rho' \in [\mathcal{H}(s_a^{h'})]$ such that $\neg(\rho \prec_a^{h\hat{}h'} \rho')$, *i.e.* $\neg(h'\hat{}\rho \prec_a^h h'\hat{}\rho')$. So $h'\hat{}\rho \in \gamma_a(h, s_a)$ since $h'\hat{}\rho' \in [\mathcal{H}(s_a)] \subseteq \gamma_a(h, s_a)$.
2. Let $\rho \in \gamma_a(h\hat{}h', s_a^{h'}) \setminus \gamma_a(h\hat{}h', s'_a)$, so $\rho \prec_a^{h\hat{}h'} \rho'$ for all $\rho' \in \gamma_a(h\hat{}h', s'_a)$ by Definition 6.
3. Let $\rho \in \gamma_a(h\hat{}h', s'_a)$, so $h'\hat{}\rho \in h'\hat{}\gamma_a(h\hat{}h', s_a^{h'})$ by assumption, so $h'\hat{}\rho \in \gamma_a(h, s_a)$ by Lemma 9.1.

□

Lemma 10 below assumes a uniform bound for the memory size and infers information about the best guarantee.

Lemma 10. Let g be a game on a finite graph with strict weak order $\prec_a$ for some $a \in A$, and let $m \in \mathbb{N}$. In each of the threshold games for a of the future games let us assume that

1. if Player 1 has a winning strategy, she has one with memory size m. Then for all $h \in \mathcal{H}$ there exists a strategy s_a with memory size m such that $\gamma_a(h, s_a) = \Gamma_a(h)$.
2. one of the players has a winning strategy with memory size m. Then $\Gamma_a(h)$ has a regular $\prec_a^h$-minimum for all $h \in \mathcal{H}$.

Proof. 1. For all $\rho \in [\mathcal{H}_{g^h}]$, we have $\rho \notin \Gamma_a(h)$ iff Player 1 has a winning strategy in the threshold game for a and ρ in g^h. In this case let s_a^ρ be a winning strategy with memory size m. For a game with n vertices there are at most $(n2^m)^{(n2^m)}$ strategy profiles using m bits of memory (by the σ representation), so the s_a^ρ are finitely many, so at least one of them, which we name s_a, wins the threshold games for all $\rho \notin \Gamma_a(h)$. This shows that $\gamma_a(h, s_a) \subseteq \Gamma_a(h)$, so equality holds.
2. Towards a contradiction let us assume that $\Gamma_a(h)$ has a no $\prec_a^h$-least element, and let $\rho \in \Gamma_a(h)$, so there exists a $\prec_a^h$-smaller $\rho' \in \Gamma_a(h)$. Since a has no winning strategy for the threshold game for ρ' (of the future game at h), the determinacy assumption implies that the coalition of her opponents has one with memory size m. These strategies are finitely many, so one of them, s_{-a}, wins the threshold game for all $\rho \in \Gamma_a(h)$. So $\mathcal{H}_{g^h}(s_{-a}) \cap \Gamma_a(h) = \emptyset$, which contradicts Lemma 10.1. Furthermore, the run induced by s_{-a} and one finite-memory s_a from Lemma 10.1 is regular.

□

Lemma 11 below assumes automatic-piecewise prefix linearity and infers information about the best guarantee. For its statement to make sense, recall Observation 2.2.

Lemma 11. Let g be a game on a finite graph with automatic piecewise prefix-linear strict weak order $\prec_a$ for some $a \in A$. If $h, h' \in H \in \overline{\mathcal{H}}$, then $\gamma_a(h, s_a) = \gamma_a(h', s_a)$ for all strategies s_a for a in g^h, and $\Gamma_a(h) = \Gamma_a(h')$.

Proof. By definition of the future games and automatic-piecewise prefix-linearity $\rho \prec_a^h \rho'$ iff $h\hat{}\rho \prec_a h\hat{}\rho'$ iff $h'\hat{}\rho \prec_a h'\hat{}\rho'$ iff $\rho \prec_a^{h'} \rho'$, so $\prec_a^h = \prec_a^{h'}$. □

Lemma 12 below assumes both automatic-piecewise prefix linearity and a uniform bound for the memory size.

Lemma 12. Let g be a game on a finite graph with automatic-piecewise prefix-linear strict weak order $\prec_a$ for some $a \in A$. Let $m \in \mathbb{N}$ and assume that the threshold games for a of the future games are determined *via* size-m strategies. Let $H \in \overline{\mathcal{H}}$.

1. There exists a strategy s_a^H with memory size m such that $\gamma_a(h, s_a^H) = \Gamma_a(h)$ for all $h \in H$.
2. There exists a size-m strategy s_{-a}^H for Player 2 in g^H (*i.e.* g^h for any $h \in H$) that is winning the threshold game for a and ρ in g^H for all $\rho \in \Gamma_a(H)$ (*i.e.* $\Gamma_a(h)$ for any $h \in H$).

Proof. 1. Lemma 10.1 provides a candidate, Lemma 11 shows that it works.
2. Let $h \in H$ and let ρ be one $\prec_a^h$-minimum of $\Gamma_a(h)$ by Lemma 10.2. Since Player 1 has no winning strategy in the threshold game for a and ρ in g^h, there exists a size-m strategy s_{-a}^H that makes Player 2 win. By Lemma 11 this strategy works also for $g^{h'}$ for all $h' \in H$.

□

Lemma 13 below already uses all the assumptions used in Theorem 5, but only for one given player. Note that if we assume the stronger Mont condition, rather than just regular-Mont, we do obtain a stronger result in Lemma 13 than we need for Theorem 5.

Lemma 13. Let g be a game on a finite graph, and let some $\prec_a$ be an automatic-piecewise prefix-linear regular-Mont strict weak order with k pieces. Let $m \in \mathbb{N}$ and assume that the threshold games for a of the future games are determined *via* size-m strategies. There is a strategy s in g such that $\mathrm{Reg} \cap \gamma_a(h, s^h) = \mathrm{Reg} \cap \Gamma_a(h)$ for all $h \in \mathcal{H}$, and that uses $m + 2\log k$ bits of memory, where Reg denotes the set of all regular runs $\rho \in [\mathcal{H}]$.

If $\prec_a$ is as above, but even fulfills the Mont condition, then we can ensure $\gamma_a(h, s^h) = \Gamma_a(h)$ for all $h \in \mathcal{H}$.

Proof. We define a strategic implementation for s in pseudocode in Algorithm 1. The algorithm uses in particular that by Lemma 12.1 for any piece $\overline{h}$ there is a strategic implementation using m bits for a strategy $s_a^{\overline{h}}$ such that $\gamma_a(h, s_a^{\overline{h}}) = \Gamma(h)$ for all $h \in \overline{h}$. An index of one of these strategic implementations is always stored, and the combined strategic implementation then follows the stored strategy as long as this one continues to realize the guarantee. If this is no longer the case, all of the k strategic implementations are checked, and one is chosen that does realize the guarantee, and this one is followed from there onwards.

We proceed to prove that the strategy implemented by the Algorithm 1 indeed satisfies our criteria. Let $h_0 \in \mathcal{H}$. To show that $\gamma_a(h_0, s^{h_0}) = \Gamma_a(h_0)$ ($\mathrm{Reg} \cap \gamma_a(h_0, s^{h_0}) = \mathrm{Reg} \cap \Gamma_a(h_0)$), let $\rho \in \mathcal{H}(s^{h_0})$ ($\rho \in \mathrm{Reg} \cap \mathcal{H}(s^{h_0})$) and let us make a case distinction. First case, a changes strategies finitely many times along ρ. Let $h_1, \dots, h_n$ be such that for all $1 \leq i \leq n$ the i-th update along ρ occurs at history $h'_i := h_0\hat{}h_1\hat{}\dots\hat{}h_i$. Applying Lemma 9.3 n times yields $h_1\hat{}\dots\hat{}h_n\hat{}\gamma_a(h'_n, t_{h'_n}) \subseteq \dots \subseteq h_1\hat{}\gamma_a(h'_1, t_{h'_1}) \subseteq \gamma_a(h_0, t_{h_0}) \subseteq \Gamma_a(h_0)$. So $\rho \in \Gamma_a(h_0)$ since $\rho \in h_1\hat{}\dots\hat{}h_n\hat{}\gamma_a(h'_n, t_{h'_n})$.

Second case, a changes strategies infinitely many times along ρ. Let $(h_n)_{n\geq 1}$ be the paths in (V, E) such that the n-th change occurring strictly after h_0 occurs at history $h'_n := h_0\hat{}h_1\hat{}\dots\hat{}h_n$. By Lemmas 10.2 and 11, for all $H \in \overline{\mathcal{H}}$ let ρ_a^H be a regular $\prec_a^H$-minimum of $\Gamma_a(H)$. By Lemma 9.2, for all $1 \leq n$ there exists $\rho' \in \gamma_a(h'_{n+1}, t_{h'_n}^{h_{n+1}})$ such that $\rho' \prec_a^{h'_{n+1}} \rho''$ for all $\rho'' \in \gamma_a(h'_{n+1}, t_{h'_{n+1}})$, especially $\rho' \prec_a^{h'_{n+1}} \rho_a^{\overline{h'_{n+1}}}$. Since $h_{n+1}\rho' \in \gamma_a(h'_n, t_{h'_n})$ by Lemma 9.1, we find $\rho_a^{\overline{h'_n}} \prec_a^{h'_n} h_{n+1}\hat{}\rho_a^{\overline{h'_{n+1}}}$. By finiteness of $\overline{\mathcal{H}}$ one $H \in \overline{\mathcal{H}}$ occurs infinitely many times as a $\overline{h'_n}$. For all $n \geq 1$ let $h'_{\varphi(n)}$ be the n-th corresponding history.

If ρ is regular, there is some $f : \mathbb{N} \to \mathbb{N}$, finite path h' and cycle h such that $h_0\hat{}h'\hat{}h^{[n]} = h'_{\varphi(f(n))}$ (and thus $h'\hat{}h^{[\omega]} =$

```
Data: Current local strategy implementation Strat
current local memory content Mem
current piece Piece
Function updatePiece is
    input : A piece H of history and a vertex v ∈ V
    output: The piece of history after H and then v
end
Function updateLocal is
    input : a strategy implementation s, a local memory
            content M, the vertex v the play is arriving in
    output: the updated local memory content M' and the
            vertex v' the strategy s wants to move to
end
Function realizesGuarantee is
    input : a strategy implementation s, a local memory
            content M, the current piece H
    output: a boolean answer whether (s, M) is realizing the
            guarantee at H
end

Function Strategy is
    input : vertex v the play is arriving in
    output: vertex v' the strategy wants to move to
    Piece:= updatePiece (Piece, v);
    (M', v') := updateLocal (Strat, Mem, v);
    if realizesGuarantee (Strat, M', Piece) then
        Mem:= M';
        return v';
    end if
    else
        foreach Strategy implementation s do
            (M',v') := updateLocal (s, 0^m, v);
            if realizesGuarantee (s, M', Piece) then
                Strat:= s;
                Mem:= M';
                return v';
            end if
        end foreach
    end if
end
```

Algorithm 1: Strategy for player a realizing guarantees

ρ). The inequality above can then be rewritten $h_0\hat{}h'\hat{}h^{[n]}\hat{}\rho_a^{\overline{h}} \prec_a h_0\hat{}h'\hat{}h^{[n+1]}\hat{}\rho_a^{\overline{h}}$ for all $n \geq 1$, so $h_0\hat{}h'\hat{}\rho_a^{\overline{h}} \prec_a h_0\hat{}\rho$ by the regular-Mont condition, so $h'\hat{}\rho_a^{\overline{h}} \prec_a^{h_0} \rho$.

In the general case, let $h = h'_{\varphi(1)}$, and let h' be such that $h = h_0\hat{}h'$. The inequality above can then be rewritten $h'_{\varphi(n)}\hat{}\rho_a^{\overline{h}} \prec_a h'_{\varphi(n+1)}\hat{}\rho_a^{\overline{h}}$ for all $n \geq 1$, so $h\hat{}\rho_a^{\overline{h}} \prec_a h_0\hat{}\rho$ by the Mont condition, so $h'\hat{}\rho_a^{\overline{h}} \prec_a^{h_0} \rho$.

Since $h'\hat{}\rho_a^{\overline{h}} \in \Gamma_a(h_0)$ by the finite case above, $\rho \in \Gamma_a(h_0)$.

Let us now analyze the pseudo-code in Algorithm 1 and find out how much memory suffices. The algorithm keeps track of the current piece H of history so far, which by assumption requires $\log k$ bits. It also keeps track of an index of the current strategic implementation, which again requires $\log k$ bits (as we need at most one strategic implementation per piece). Finally, we use m bits for the memory content M. □

Proof of Theorem 5. We combine Lemmas 10.1 and 13 to obtain a finite memory strategy s_a for each player a such that Reg $\cap$ $\gamma_a(h, s_a^h) =$ Reg $\cap\, \Gamma_a(h)$ for all $h \in \mathcal{H}$. The run ρ_{NE} induced by this strategy profile will be the run induced by the Nash equilibrium. As ρ_{NE} is induced by a finite-memory strategy profile, $\rho_{NE} \in$ Reg, and in particular, for any decomposition $\rho_{NE} = h\hat{}\rho$ and any player we find that $\rho \in \Gamma_a(h)$. Now we need to ensure that no one has any incentive to deviate.

For all $a \in A$ and all pieces $H \in \overline{\mathcal{H}}$ of histories ending in V_a let s_{-a}^H be the strategy from Lemma 12.2. For all $a \in A$ let a play as follows: Keep playing according to ρ_{NE} until one player b deviates from ρ_{NE} at some history h_D. If $b = a$, *i.e.*, the last vertex of h_D is in V_a, let a do whatever; otherwise let a play according to $s_{-b}^{\overline{h_D}}$ in g^{h_D}, *i.e.* let a take part in the one-vs-all coalition that ensures that b cannot obtain anything $\prec_a^{h_D}$-better than $\rho_{\overline{h_D}}$ in g^{h_D}, so $\neg(\rho_{\overline{h_D}} \prec_b^{h_D} \rho_D)$, where ρ_D is the new run in g^{h_D} after deviation by b. Let $h_D\hat{}\rho = \rho_{NE}$. By construction of ρ_{NE} and Lemma 13, $\rho \in \Gamma_b(h_D)$, so $\neg(\rho \prec_b^{h_D} \rho_{\overline{h_D}})$. So $\neg(\rho_{NE} \prec_b h_D\hat{}\rho_D)$ since $\prec_b$ is a strict weak order, *i.e.* b has no incentive to perform a deviation.

Each player a needs to run the intended strategies s_b for all other players b, too, in order to be able to detect deviation. This requires $|A|(m + 2\log k)$ bits by Lemma 13. If a deviation is detected, the punishment strategy for that history is executed instead. (Note that the history where the deviation happened includes the information on who deviated; and that the players already keep track of the history in the strategy of Lemma 13.) The punishment phase requires $\log k$ memory to record which strategy to follow, and m bits to follow it. However, the original memory can be repurposed, by using just one extra bit to determine whether deviation ever occurred. This yields the overall memory bound $|A|(m + 2\log k) + 1$. □

Comparison to previous work

As mentioned above, a similar but weaker result (compared to our Theorem 5) has previously been obtained by BRIHAYE, DE PRIL and SCHEWE (Brihaye et al. 2013),(Pril 2013, Theorem 4.4.14). They use cost functions rather than preference relations. Our restriction to strict weak orders is strictly more general[2]. However, even if both frameworks are available, it is more convenient for us to have results formulated via preference relations rather than cost functions: Cost functions can be translated immediately into preferences, whereas translating preferences to cost functions is more cumbersome. In particular, it can be unclear to what extend *nice* preferences translate into *nice* cost functions. Note also that prefix-linearity for strict weak orders is more general than prefix-linearity for cost functions.

As a second substantial difference, (Pril 2013, Theorem 4.4.14) requires either prefix-independent cost functions and finite memory determinacy of the induced games, or prefix-linear cost functions and positional subgame-perfect strategies. This, essentially, means that they assume the result of Lemma 13. In particular,(Pril 2013, Theorem 4.4.14) cannot be applied to energy parity games, where finite prefixes of the run do impact the overall value for the players, and where at least the protagonist requires memory to execute a winning strategy.

The (regular) Mont condition is absent in (Brihaye et al. 2013), but it can be shown that their other requirements imply the regular-Mont condition.

Before (Pril 2013; Brihaye et al. 2013), it had already been established by PAUL and SIMON (Paul and Simon 2009) that multi-player multi-outcome Muller games have Nash equilibria consisting of finite memory strategies. As (two-player win/lose) Muller games are finite memory determined (Gurevich and Harrington

[2] For example, the lexicographic combination of two payoff functions can typically not be modeled as a payoff function, as $\mathbb{R} \times \{0, 1\}$ (with lexicographic order) does not embed into $\mathbb{R}$ as a linear order.

1982), and the corresponding preferences are obviously prefix independent, this result is also a consequence of (Pril 2013, Theorem 4.4.14). Another result subsumed by (Pril 2013, Theorem 4.4.14) (and subsequently our main theorem) is found in (Brihaye et al. 2010) by BRIHAYE, BRUYÈRE and DE PRIL.

Algorithmic considerations

Let us briefly consider the algorithmic price to pay for the extension for the two-player win/lose case to the multi-player multi-outcome situation. Let us assume that for some class of games satisfying the criteria of Theorem 5, computing a winning strategy of a two-player win/lose game of size n has winning strategies takes $f(n)$ time. Let us further assume that, given finite memory strategies of size up to m bits for each player, we can decide in time $g(n,m)$ who is winning. Additionally, let us assume that a multi-player multi-outcome game of size n induces up to $h(n)$ one-vs-all threshold games of size n.

In Lemma 10.1, we are essentially proceeding by brute force: We are investigating up to $h(n)$ induced one-vs-all threshold games, and are asking for a winning strategy in each of them, which could take up to $h(n)(f(n)+g(n,m))$ time. In any concrete example, though, a much more efficient construction is to be expected – for example, Theorem 30 below provides such a construction of the induced one-vs-all threshold games coming from multi-player multi-outcome energy parity games.

In Lemma 13, we need to invoke Lemma 10.1 for each combination of memory content (2^m different values) and piece of the partition (k). Thus, we are spending up to $2^m kh(n)(f(n)+g(n,m))$ time to obtain sufficiently many strategies to realize the guarantee everywhere, which we can combine in linear time to yield the single strategy output strategy.

In the proof of Theorem 5, we invoke Lemma 13 once per player, which costs $O(|A|2^m kh(n)(f(n)+g(n,m))$ time. In addition, we need k many winning strategies in an induced one-vs-all threshold game, for additional cost of $O(kf(n))$ time – in total, we are still at $O(|A|2^m kh(n)(f(n)+g(n,m))$.

Let us introduce some realistic but simplifying additional assumptions. We would expect h to be of the form $2^{O(n)}$. The parameter k will be dominated by $2^{O(n)}$ in most situations. Typical finite-memory determined win/lose games might required exponential memory (if measured by number of states), but as we measure m in bits, also 2^m would be absorbed into $2^{O(n)}$. By dropping the distinctions between finding the winning strategy and determining who is winning, and noting that $|A| \leq n$, we arrive at an overall complexity of $2^{\mathcal{O}(n)}f(n)$. The additional factor of 2^n will in some cases worsen the asymptotic runtime significantly (e.g. for parity games, subexponential algorithms are known (Jurdzinski et al. 2008)), but in others, pales against the complexity for solving the two-player win/lose case.

On automatic-piecewise prefix-linearity

While the definition of automatic-piecewise prefix-linearity may seem a bit complicated at first glance, the notion seems to fit in well with finite memory strategies: Intuitively, the requirement is merely that however we split a run into a finite prefix and an infinite tail, the contribution of the prefix to the value for the player factors via some fixed (i.e. independent of the length of the prefix) number, and does so via a finite automaton. Whenever this is not satisfied, one would expect that in principle a finite memory strategy may fail (compared to an unrestricted strategy), simply because it cannot properly account for the contribution of the finite prefix it has seen so far.

Of the popular winning conditions, many are actually prefix-independent, such as parity, Muller, mean-payoff, cost-Parity, cost-Streett, etc. Clearly, any combination of prefix-independent conditions itself will be prefix-independent. Typical examples of non-prefix independent conditions are reachability, energy, first-cycle (Aminof and Rubin 2014) and discounted payoff. We can easily verify that combining a reachability or energy condition with any prefix-linear condition yields an automatic-piecewise prefix-linear condition (provided that energy is bounded).

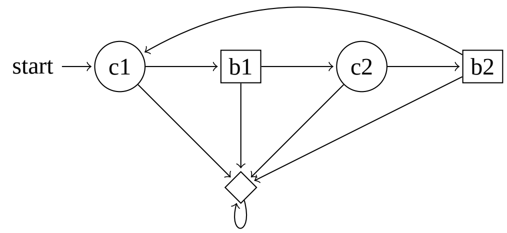


Figure 2: The graph for the game in Example 14

Discounted payoff can be problematic, though. Here each vertex is assigned a payoff a_v, a discount factor $\delta \in (0,1)$ is chosen, and the value of a run $\rho = v_0 v_1 \dots$ is $\sum_{i=0}^{\infty} v_i \delta^i$. While discounted payoff on its own is of course prefix-linear, it does not combine well with other criteria: For example, in a generalized discounted payoff parity game the question whether we prefer a tail with a better discounted payoff but worse least priority to a tail with worse discounted payoff but better least priority may depend on the precise value of the payoff obtained in the history so far, as well as the length of the history (as later contributions to payoff count less). This is too much information for a finite automaton to remember, thus, generalized discounted payoff parity games do not satisfy the criterion for being automatic-piecewise prefix-linear.

On uniform finite memory determinacy

The requirement in Theorem 5 that there is a uniform memory bound sufficient for all threshold games is not dispensible. Mean-payoff parity games, for example, satisfy all other criteria, yet lack finite memory Nash equilibria, as the following example shows.

Let g be the one-player game in Figure 1b. The payoff of a run that visits the vertex g infinitely often is the limit (inferior or superior) of the average payoff. It is zero if g is visited finitely many times only. For any threshold $t \in \mathbb{R}$, if $t < 1$, the player has a winning finite-memory strategy: cycle p times in b, where $p > \frac{1}{1-t}$, visit g once, cycle p times in b, and so on. If $t \geq 1$, the player has no winning strategy at all. So the thresholds games of g, and likewise for the future game of g, are finite-memory determined. The game has no finite memory Nash equilibrium nonetheless, since the player can get a payoff as closed to 1 as she wants, but not 1.

Note that the preceding example could also be used for discounted-payoff parity games. As discussed above though, these also fail the automatic-piece prefix-linearity condition.

On the Mont condition

We can exhibit a prototypic example for how failure of the regular-Mont condition translates into the absence of Nash equilibria:

Example 14 ([3]). The game g in Figure 2 involves Player 1 (2) who owns the circle (box) vertices. Who owns the diamond is irrelevant. The payoff for Player 1 (2) is the number of visits to a box (circle) vertex, if this number is finite, and is -1 otherwise. Let s_1 be the

[3] This example is based on an example communicated to the authors by Axel Haddad and Thomas Brihaye, which in turn is based on a construction in (Brihaye et al. 2014).

positional strategy where Player 1 chooses $b1$ when in $c1$ and the diamond when in $c2$, and let t_2 be the positional strategy where Player 2 always choses the diamond. With s_1 Player 1 secures payoff 1, and with t_2 Player 2 makes sure that Player 1 is not getting more than that. Let s_2 be any positional strategy for Player 2, and let t_1 be the positional strategy where Player 1 always choses the diamond. With s_2 Player 1 secures payoff 1, and with t_1 Player 1 makes sure that Player 2 is not getting more than that. Therefore the threshold games of g are positionally determined, and likewise for the future games of g. The game g has no Nash equilibrium nonetheless: in the run induced by a putative NE, one of the players has to choose the diamond at some point (to avoid payoff -1), but by postponing this choice to next time, the player can increase her payoff by 1. This shows the relevance of the Mont condition.

4. Equivalence between energy Muller games and bounded-energy muller games

Energy games were first introduced in (Chakrabarti et al. 2003): Two players take turns moving a token through a graph, while keeping track of the *current energy level*, which will be some integer. Each move either adds or subtracts to the energy level, and if the energy level ever reaches 0, the protagonist loses. These conditions were later combined with parity winning conditions in (Chatterjee and Doyen 2012) to yield energy parity games as a model for a system specification that keeps track of gaining and spending of some resource, while simultaneously conforming to a parity specification.

In both (Chakrabarti et al. 2003) and (Chatterjee and Doyen 2012) the energy levels are a priori not bounded from above. This is a problem for the applicability of Theorem 5, as unbounded energy parity preferences are not automatic-piecewise prefix-linear in the general case. In (Bouyer et al. 2008), two versions of bounded energy conditions were investigated: Either any energy gained in excess of the upper bound is just lost (as in e.g. recharging a battery), or gaining energy in excess of the bound leads to a loss of the protagonist (as in e.g. refilling a fuel tank without automatic spillover prevention). In the following, we only consider the former[4]. As a finite automaton can easily keep track of the energy level between 0 and the upper bound, bounded energy parity preferences are automatic-piecewise prefix linear.

Definition 15 (2PWLuEP game)**.** A two-player win/lose (unbounded) energy parity game (2PWLEP game) is given by a graph (V, E) with a vertex partition $V = V_2 \cup V_2$, and starting vertex v_0, an energy threshold $E_{\min}$ and labels associating two integers to each vertex v: a priority π_v and an energy delta δ_v.

Given a run $\rho = v_0v_1\ldots$, we consider two values: The least priority π occurring infinitely many times, and the least cumulative energy value ever reached, $E = \min_{n\in\mathbb{N}} \sum_{i=0}^{n} \delta_{v_i}$ (this may be $-\infty$). Player 1 wins, if π is odd and $E > E_{\min}$.

Definition 16 (2PWLbEP game)**.** In the bounded variant, there is additionally an upper energy bound $E_{\max}$. The cumulative energy values E_n in a run $\rho = v_0v_1\ldots$ are instead calculated as $E_0 = \delta_{v_0}$, $E_{n+1} = \min\{E_{\max}, E_n + \delta_{v_{n+1}}\}$. Again, the least priority π occurring infinitely many times together with $E = \min_{n\in\mathbb{N}} E_n$ determine the winner. Player 1 wins, if π is odd and $E > E_{\min}$.

Typically, an initial starting energy is either considered part of the game, or the question becomes whether some value exists letting Player 1 win. As shown in (Chatterjee and Doyen 2012) the latter reduces to the former by picking a sufficiently large value. In our formalism we can simulate an arbitrary initial credit by letting v_0 have no incoming edges and the desired initial energy credit as δ_{v_0}. We would then add an ersatzv_0', and extend the preferences to take both v_0 and v_0' into account.

In this section we will show that, as far as two-player win/lose games are concerned, bounded energy parity games and unbounded energy parity games are equivalent, in the sense that introducing a sufficiently large upper bound on the energy will not change which player has a winning strategy. Actually, we will prove the corresponding result for the more general case of energy Muller games first. In these, just the parity condition in the definitions above is replaced with a Muller condition specifying which sets of vertices occurring infinitely many times are *good* for Player 1.

Our proof combines a repeated surgery argument on a winning strategy with a combinatorial observation on a class of finite games related to Muller games:

Definition 17. An M-reachability game is given by a graph (V, E) partitioned into two sets $V = V_2 \cup V_2$, a starting vertex v_0, a set of target vertices W and a set of sets of vertices $\mathfrak{V}$. Player 1 wins iff the run reaches some $v \in W$ and the set of vertices visited along the way falls into $\mathfrak{V}$.

Theorem 18. If Player 1 can win an M-reachability game played in a graph with $|V| = n$, then he can do so in at most $\frac{n(n+1)}{2}$ steps (i.e. has a winning strategy that guarantees a winning prefix of length no more than $\frac{n(n+1)}{2}$).

Proof. Omitted (see Section A in the appendix). ☐

Even in an M-reachability game where only Player 1 moves at all, he may need to make $\frac{n(n+1)}{4}$ steps. The example showing this is the same as the one employed for Remark 25 below. While the gap between our upper and lower bound is only a factor 2, it would nevertheless be interesting to identify the precise value of the time needed.

Open Question 19. What is the precise value of the number of steps needed by Player 1 to win an M-reachability game that he indeed can win?

Theorem 20. If a player has a winning strategy in an energy Muller game g, he or she also wins the $\frac{n(n+1)}{2}W$ bounded variant g' of g, where W is the maximal module of the negative deltas occurring in g.

A winning strategy for Player 2 in g' is also winning in g. Let s_0 be a winning strategy for Player 1 in g. Below we modify s_0 recursively such that the limit strategy is winning in g'.

Note that the only reason why s_0 might fail to be winning in g' while winning in g is that it could accumulate more than $\frac{n(n+1)}{2}W$ energy, and then spent more than $\frac{n(n+1)}{2}W$ (but less than the accumulated amount). We will show that in these situations, the accumulation and spending phases can be distributed better, and that this will induce a winning strategy in g'. The relevant obstructions are defined as follows:

Definition 21. We consider histories $h \in \mathcal{H}(s_0)$ where the current energy level is locally maximal (i.e. at the step prior to h, the energy level was less, and there is a consistent extension by one move not increasing the energy). Let $B \subseteq \mathcal{H}(s_0)$ be the set of shortest consistent histories extending h such that for each $h' \in B$ the lowest energy level reachable in some consistent extension of h' is already reached at h'. If the energy level between h and the $h' \in B$ with the least energy level decreases by more than $\frac{n(n+1)}{2}W$, we call such a history h a *bad history* (w.r.t. s_0).

As s_0 is a winning strategy, the B as above is a bar (i.e. any consistent run extending h also extends some $h' \in B$). By compactness, then, we find that B is finite.

[4] With the exception of Corollary 42 in Section 6.

Lemma 22. For a winning strategy s of Player 1 in g exactly one of the following holds:

1. s is also winning in g'.
2. There is a bad history $h \in \mathcal{H}(s)$.

Proof. Omitted (see Section B in the appendix). □

Let W be the set of vertices occurring as the last positions of histories $h' \in B$, and let v be the last vertex of h. Clearly, Player 1 can force the run to reach W from v. Moreover, for each $h' \in B$ let $V_{h'}$ be the set of vertices occurring between h and h', and let $\mathfrak{V} = \{V_{h'} \mid h' \in B\}$. Now player 1 can even force the run to reach W from v such that the set of vertices used falls into $\mathfrak{V}$. Thus, by Theorem 18 we find that Player 1 can achieve this within $\frac{n(n+1)}{2}$ moves.

Definition 23. Now we let $s^{\star h}$ work as follows: Starting playing according to s. If h is ever reached, switch to the fast winning strategy for the induced M-reachability game. Once some $u \in W$ is reached, let $V_u \in \mathfrak{V}$ be vertices that have been visited from h till now, and switch back to s while pretending that some $h' \in B$ happened with u as the last vertex and such that $V_u = V_{h'}$.

Then we inductively define s_0' as follows: Let $s_0^0 := s_0$. If s_0^i has a bad history, let h be a shortest one (use lexicographic order to resolve ambiguity), and then let $s_0^{i+1} := (s_0^i)^{\star h}$. Else let $s_0^{i+1} := s_0^i$. Let $s_0' := \lim_{i\to\infty} s_0^i$.

Lemma 24.
1. For a winning strategy s (in g), also $s^{\star h}$ is winning.
2. h is not bad w.r.t. $s^{\star h}$.
3. If h' does not extend h and h' is not bad w.r.t. s, then h' is not bad w.r.t. $s^{\star h}$.
4. If $\lim_{i\to\infty} s^i = s$ and h is bad w.r.t. s, then for all but finitely many i, h is bad w.r.t. s^i.

Proof.
1. First we argue that $s^{\star h}$ meets the energy constraints. As $s^{\star h}$ coincides with s up to h, and by definition of a bad history h, the energy level at h is guaranteed to be at least $\frac{n(n+1)}{2}W$. As the part of $s^{\star h}$ taken from the quick winning strategy in the induced M-reachability game takes at most $n(n+1)$ moves, the energy level after that is guaranteed to be non-negative. As the histories in B have the property that following s from them will never drop beneath the original value, playing according to $s^{\star h}$ will then never reach negative energy levels.
   As we alter only a finite prefix of the strategy, the Muller condition also remains satisfied.
2. Energy levels are affine w.r.t. additive changes of the initial energy level. Thus, as $s^{\star h}$ coincides with s from any history in B onwards, the energy levels obtained with $s^{\star h}$ differ from those obtained with s by the same amount as the two strategies differ for histories in B. In particular, the property that the histories in B are the lowest energy levels ever reached after them is preserved. But by construction of $s^{\star h}$, the energy levels for histories in B are at most $\frac{n(n+1)}{2}W$ below the one at h, thus h is not a bad history w.r.t. $s^{\star h}$.
3. If h and h' are incomparable, then s and $s^{\star h}$ coincide on any run through h'. Let us now assume that h' is a prefix of h. Then the energy level at h' is the same for s and $s^{\star h}$. If the energy level at h' is above $\frac{n(n+1)}{2}W$, then h' was already bad w.r.t. s, as the histories in B can also be reached from h'. If the energy level at h' is at or below $\frac{n(n+1)}{2}W$, it cannot be bad anyway.
4. Being bad depends only on the strategy up to the histories in B, which constitutes a finite prefix of s by König's Lemma.

□

Proof of Theorem 20. It only remains to show that s_0' as constructed above is a winning strategy for Player 1 in g'. First, we argue that s_0' is a winning strategy for Player 1 in g. As meeting the energy constraints is a closed property (and thus preserved by taking limits), Lemma 24 (1) implies that s_0' meets the energy constraint.

Towards a contradiction, let us assume that Player 2 can falsify the Muller winning condition against s_0'. In the resulting play, let us consider the tails starting where some bad history was removed in the construction. All but finitely many of these need to visit some set of vertices not in the winning condition. However, by as we modify s_0 by winning strategies in the M-reachability game to obtain s_0', we find that Player 2 could achieve the same sets of vertices visited in the corresponding tails already against s_0 – but then s_0 is no winning strategy either.

Now, by Lemma 24 we can conclude that s_0' has no bad histories. Then Lemma 22 implies that s_0' is winning in g'. □

Remark 25. The quadratic bound from Theorem 20 above is tight up to a factor of 2.

Proof. For all $n \in \mathbb{N}$ let G_n be the graph with vertices $x_1, \dots, x_n$, $y_1, \dots, y_n$ with the arcs (x_i, x_{i+1}) for all possible i, and (x_n, y_i) and (y_i, x_1) for all i. Every path in G_n that involves all the vertices has length at least $n(n+1)$. □

Theorem 26. If a player has a winning strategy in an energy parity game g, he or she also wins the $2nW$ bounded variant g' of g.

Proof. As parity games are a special case of Muller games, we only need to argue that the required energy bounds are lower. The value $\frac{n(n+1)}{2}W$ in the statement of Theorem 20 comes from Theorem 18, so it suffices to argue that if the vertex sets in an M-reachability game are derived from a parity condition, then $2n$ steps are enough to win it (if it can be won). This in turn works by first going to some high winning priority (in at most n steps), and then to some vertex in $\mathfrak{V}$ (in at most n steps). □

Remark 27. Example 38 also shows that the restriction to Muller conditions is not entirely dispensable in Theorem 20.

5. Multi-player multi-outcome energy parity games

There is one obstacle left preventing us from combining Theorem 5 with the finite memory determinacy of energy parity games from (Chatterjee and Doyen 2012) to derive finite memory Nash equilibria for multi-player multi-outcome energy parity games (MMEP games). The one-vs-all threshold future games induced by MMEP games will have disjunctions of winning conditions for the first player in an energy parity game as objectives – and classical energy parity conditions (i.e. those in 2PWLuEP or 2PWLbEP games, Definitions 16, 15) are not closed under disjunction.

We will first formally define MMEP games, and then generalized two-player win/lose energy parity games. We prove finite memory determinacy of the latter, which also implies a proof of finite memory determinacy of (classical) energy parity games. Our proof is much simpler (as most of the difficulty is delegated to Theorem 26) and provides better bounds on the memory required contrasted to the ones in (Chatterjee and Doyen 2012). We will also discuss its algorithmic implications at the end of the section.

Definition 28. A multi-player multi-outcome energy parity games (MMEP game) is played on a graph (V, E) by some finite set of players from some starting vertex v_0. The vertex set is partitioned in some $V \bigcup_{a\in A} V_a$. For each combination of player a and vertex

v there is some energy delta δ^a_v and priority π^a_v. Each player has an upper energy bound $E^a_{\max}$.

The cumulative energy values E^a_n for player a in a run $\rho = v_0 v_1 \ldots$ are calculated as $E^a_0 = \delta^a_{v_0}$, $E^a_{n+1} = \min\{E^a_{\max}, E^a_n + \delta^a_{v_{n+1}}\}$. Player a only cares about the least priority π^a occurring infinitely many times together with $E^a = \min_{n\in\mathbb{N}} E^a_n$. He has some strict weak order $\prec_a$ on such pairs, which respects $E < E' \Rightarrow (\pi, E') \not\prec_a (\pi, E)$.

Definition 29. A generalized two-player win/lose energy parity game (G2PWLEP game) is given by a graph (V, E) with a vertex partition $V = V_1 \cup V_2$, and starting vertex v_0, an energy threshold $E_{\min}$ and labels associating two integers to each vertex v: a priority π_v and an energy delta δ_v.

Given a run $\rho = v_0 v_1 \ldots$, we consider two values: The least priority π occurring infinitely many times, and the least cumulative energy value reached, $E = \min_{n\in\mathbb{N}} \sum_{i=0}^{n} \delta_{v_i}$. The winning condition is given by some family $(E^i_{\min}, P_i)_{i\in I}$ of energy thresholds and sets of priorities. Player 1 wins iff there is some $i \in I$ with $E > E^i_{\min}$ and $\pi \in P_i$. We will write $B_{\min} = \min_{i\in I} E^i_{\min}$.

A 2PWLbEP game is clearly the special case of a G2PWLEP game with $|I| = 1$. As Theorem 26 shows that 2PWLuEP games may be considered as a special case of 2PWLbEP games, we find G2PWLEP games to indeed be a generalization of the other two-player win/lose versions of energy parity games, thus deserving their names. It is also easily verified that each one-vs-all threshold future game induced by a MMEP game is a G2PWLEP game.

Theorem 30. G2PWLEP games are finite memory determined, and $\log|E_{\max} - B_{\min}|$ of memory suffices for Player 1, and $2\log|E_{\max} - B_{\min}|$ for Player 2.

Proof. We consider the expansion by the set $Q := \{B_{\min}, \ldots, E_{\max}\} \times \{B_{\min}, \ldots, E_{\max}\}$, where there is an edge from (v, e_0, e_1) to (u, e'_0, e'_1) iff there an edge from v to u in the original graph, $e'_0 := \min\{E_{\max}, \max\{B_{\min}, e_0 + \delta_v\}\}$ and $e'_1 = \min\{e_1, e'_0\}$. Essentially, we keep track of both the current energy level and of the least energy level ever encountered as part of the vertices. Note that there never can be an edge from some (v, e_0, e_1) to (u, e'_0, e'_1) where $e'_1 > e_1$.

Let $T_k := \{\pi \mid \exists i\ \pi \in P_i \wedge E^i_{\min} \geq k\}$. Now we first consider the subgraph induced by the vertices of the form $(v, e_0, 0)$; and then the parity game played on this subgraph with winning priorities T_0. As parity games admit positional strategies that win from every possible vertex, we can fix such strategies for both players on the subgraph.

Now we extend these strategies via reachability analysis to some maximal subgraph, and in the following, only consider the complement. In the next stage, we consider the subgraph of this complement induced by the vertices of the form $(v, e_0, 1)$, and again consider a parity game played there, this time with winning priorities in T_1.

Iterating the parity-game and reachability analysis steps will yield positional optimal strategies for both players on the whole expanded graph.

Now consider the winning sets and strategies of both players: If Player 1 wins from some vertex (v, e_0, e_1), then he also wins from any (v, e_0, e'_1) where $e_1 < e'_1$ – for the only difference is the lowest energy level ever reached, which can only benefit, but not harm, Player 1. Moreover, as (v, e_0, e'_1) cannot be reached from (v, e_0, e_1) at all, Player 1 can safely play the same vertex u in the original graph at (v, e_0, e'_1) as he plays at (v, e_0, e_1). For fixed v, e_0, let e_1 be minimal such that Player 1 wins from (v, e_0, e_1). Then we can change his strategy such that he plays the same vertex in the underlying graph from any (v, e_0, e'_1).([5])

By using $\log|E_{\max} - B_{\min}|$ bits of memory, Player 1 can play his positional strategy from above in the original game. Likewise, Player 2 can play her positional strategy from the expanded graph in the original game using $2\log|E_{\max} - B_{\min}|$ bits of memory. ☐

Corollary 31. 2PWLuEP games are finite memory determined, and $\log 2nW$ bits suffice.

Proof. By Theorem 26 we may consider 2PWLbEP games with energy bound $2nW$ instead. From Theorem 30, we obtain a winning strategy for Player 1 using $\log 2nW$ bits of memory, if he has a winning strategy at all. It is easy to see that in a 2PWLuEP game won by Player 2, she even has a positional winning strategy (as she can either play a positional strategy forcing the energy to drop beneath Player 1s threshold or a positional strategy to win the underlying parity game, plus the relevant attractors of these). ☐

The preceding corollary improves upon (Chatterjee and Doyen 2012, Theorem 1), which requires $\log 4ndW$ bits of memory, where d is the number of different priorities. If the memory cost is measured in the number of states (i.e. $2nW$ vs. $4ndW$), this actually is a noticeable difference. In particular, the fact that d disappears from the bound is due to the very different proof technique we employ here.

We need one last simple lemma, and then will be able to apply Theorem 5 to energy parity games.

Lemma 32. The valuation-preference combinations in MMEP games are automatic-piecewise prefix-linear.

Proof. Omitted (see Section B in the appendix). ☐

Corollary 33. All MMEP games have Nash equilibria in finite memory strategies. Let A be the set of players, n the size of the graph and W the largest energy delta. Further let E be the maximum difference between $E^a_{\max}$ and E^a_i for some $a \in A$, $i \in I$. Then $1 + |A| n E^{|A|} \log(2nW) + (|A|^2 + |A|) \log nE$ bits of memory suffice.

Proof. By Theorem 26 and Lemma 32 the prerequisites of Theorem 5 are given, which then yields the claim. Note that we find that $k \leq nE^{|A|}$ and $K \leq \log nE^{|A|}$ from the proof of Lemma 32, and $m = \log 2nW$ from Theorem 26. ☐

We thus see an exponential blowup compared to the two-player win/lose case in terms of size – which is disappointing if one prefers to synthesis succinct systems, but this goal is presumably unattainable in general anyway, cf. (Fearnley et al. 2012).

Algorithmic considerations

The proof of Theorem 30 immediately gives rise to an algorithm computing the winning strategies in generalized two-player win/lose energy parity game while using an oracle for winning strategies in parity games. Using e.g. the algorithm for solving parity games from (Jurdzinski et al. 2008), which has a runtime of $n^{O(\sqrt{n})}$, we obtain a runtime of $(nE)^{O(\sqrt{nE})}$, if we set $E := |E_{\max} - B_{\min}|$. Unfortunately, only the binary representation of E will need to be present in the input – E itself can easily be exponential in the size of the input.

[5] This trick does not work for Player 2, because we would need to consider the maximal e_1 where she wins (instead of the minimal one for Player 1), but then the backward induction from the middle of the proof goes in the "wrong direction".

If we consider the special case of 2PWLuEP games in Corollary 31, we find a runtime of $(nW)^{O(n\sqrt{W})}$ – which will generally be worse than the runtime $O(dn^{d+5}W)$ obtain in (Chatterjee and Doyen 2012).

If we assume W to be fixed[6], we arrive at a Cook reduction of solving G2PWLEP games to solving parity games. This in particular implies that the decision problem for G2PWLEP games with bounded weights is in UP $\cap$ co-UP. For 2PWLuEP games with bounded weights we even obtain a Karp reduction to parity games, and thus membership in PLS $\cap$ PPAD.

6. Extension of determinacy by combination

In Sections 4 and 5 we have studied games where the winning condition for Player 1 is the conjunction of a parity condition and an energy condition. Each of them guarantees memoryless determinacy, but the combined condition only guarantees finite-memory determinacy. It is yet another hint in favor of finite memory, as opposed to zero memory. In this section we generalize the above-mentioned combination in several ways: The parity condition is generalized into conditions that guarantee finite-memory determinacy and that meet some additional weak requirements; the energy condition is generalized into all the finite prefixes of the run being rejected by a (usually regular) language; and we also consider disjunctions.

The pairing of a non-trivial history of a game is defined inductively by $f(v_0v) := (v_0, v)$ and $f(hvv') := f(hv)(vv')$ for $h \in \mathcal{H}$, and it is easily extended to infinite runs $\rho \in [\mathcal{H}]$. For a run ρ let $Pref(\rho)$ be the finite prefixes of ρ. For every finite automaton $\mathcal{A}$ let $L_\mathcal{A}$ be the words accepted by $\mathcal{A}$.

Lemma 34 below involves the weakest requirements from this section, but it applies only for Player 2 and disjunctions.

Lemma 34. Let $g = \langle (V, E), v_0, V_1, V_2, W\rangle$ be a win/lose games on a finite graph such that if Player 2 can win g or its future games, she can win by using finite memory. Let L be a language over the alphabet E, and let us derive g_L from g by replacing W with W_L such that $\rho \in W_L$ iff $\rho \in W \vee f(Pref(\rho)) \cap L = \emptyset$. If Player 2 wins g_L, she has a finite-memory winning strategy.

Proof. Among the histories h in g whose pairings are in L, let $\mathcal{H}_L$ contain the minimal ones for the prefix relation. Let $\mathcal{H}_2$ be the histories h in $\mathcal{H}_L$ from where Player 2 wins the future game of g (and therefore also of g_L) after h. Let s be a winning strategy for Player 2, so every run in $[\mathcal{H}(s)]$ has a prefix in $\mathcal{H}_2$, and we can define T as the subtree (a subset that is a tree) of $\mathcal{H}(s)$ whose maximal paths are all in $\mathcal{H}_2$. By König's Lemma T is finite. For each $h \in \mathcal{H}_2 \cap T$ let s_h be a finite-memory strategy making Player 2 win the future game of g after h. The following finite-memory strategy makes Player 2 win g_L: if $h \in T \setminus \mathcal{H}_2$ go to some vertex $v \in V$ such that $hv \in T$; if h has $h_1 \in \mathcal{H}_2$ as a prefix, play as prescribed by s_{h_1}. ☐

A version of Lemma 34 for Player 1 will require the strongest assumptions, whereas the conjunctive case requires the same intermediate assumptions for Players 1 and 2. For this we need new definitions.

Definition 35 (Product of game by an automaton)**.** Let $g = \langle (V, E), v_0, V_1, V_2, W\rangle$ be a win/lose game on a graph and let $\mathcal{A} = (E, Q, q^0, F, \Delta)$ be a finite automaton. The product $g \times \mathcal{A}$ of g by $\mathcal{A}$ is the win/lose game $g' = \langle (V', E'), (v_0, q^0), V_1 \times Q, V_2 \times Q, W'\rangle$ where $((v, q), (v', q')) \in E'$ iff $(q, (v, v'), q') \in \Delta$, and $p \in W'$ iff the projection $\pi_1(p) \in W$.

[6] Which is poly-time equivalent to W being given in unary.

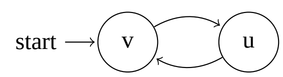


Figure 3

Definition 36 (Simple restriction of a game)**.**
Let $g = \langle (V, E), v_0, V_1, V_2, W\rangle$ be a win/lose game on a graph, and let $S \subseteq V$. If $g\ |_S := \langle (S, E \cap S^2), v_0, V_1 \cap S, V_2 \cap S, W \cap S^\mathbb{N}\rangle$ is a win/lose game on a graph, it is called the simple restriction of g to S.

If the winning condition of Player 1 is a conjunction, extra assumptions such as regularity of the language and closure properties by product and restriction yield finite-memory determinacy. Furthermore, uniformity of the memory bounds for the winning strategies transfers from assumption to conclusion.

Lemma 37. Let $g = \langle (V, E), v_0, V_1, V_2, W\rangle$ be a win/lose two-player game on a finite graph such that the simple restrictions of its product by finite automata are determined *via* strategies using finite memory (resp. $m(n)$ bits of memory, where n is the number of states). Let $\mathcal{A}$ be a finite automaton over the alphabet E, and let us derive $g_\mathcal{A}$ from g by replacing W with $W_\mathcal{A}$ such that $\rho \in W_\mathcal{A}$ iff $\rho \in W \wedge f(Pref(\rho)) \cap L_\mathcal{A} = \emptyset$. Then $g_\mathcal{A}$ is determined via strategies using finite memory (resp. $m(|V| \cdot |Q|) + \log |Q|$ bits, where Q are the states of $\mathcal{A}$).

Proof. Let $\mathcal{A} = (E, Q, q^0, F, \Delta)$ and let $g' := g \times \mathcal{A}$. Let $S \subseteq V \times Q$ be the vertices of g' from where Player 2 cannot force the run to reach $V \times F$. Let us make a case disjunction: First case, (v_0, q^0) is not in S. So in g' Player 2 can win by playing positionally. Such a strategy yields a strategy in g ($g_\mathcal{A}$) that only needs memory to run $\mathcal{A}$, in order to know the current state in Q. For this $\log |Q|$ bits are sufficient.

Second case, $(v_0, q^0) \in S$. By assumption, the simple restriction $g'\ |_S$ is determined via strategies using finite memory (resp. $m(|V| \cdot |Q|)$ bits of memory). If Player 1 has a winning strategy for $g'\ |_S$, she can use it together with $\mathcal{A}$ to play in $g_\mathcal{A}$. If Player 2 has a winning strategy, she can do the same until the run, seen as a run in g', leaves S; and then she can play as in the first case. ☐

Example 38 below shows that the regularity assumption of $L_\mathcal{A}$ in Lemma 37 is useful. Indeed the language made of the histories with positive energy levels at every prefix is not regular.

Example 38. We consider games where the vertices are colored by $\{0, 1\}$, and where Player 1 wins the runs with colors $w0^\omega$ for any $w \in \{0, 1\}^*$, as well as the runs with colors $101100 \dots 1^n 0^n \dots$. Such games are finite memory determined, but their energy-version is not.

Proof. In order to be able to win, Player 1 needs to be able to force arbitrarily long colour sequences of 0's. But if he can do that, he can force eventually constant 0 even by a positional strategy. Likewise, for Player 2 it suffices to play a positional strategy preventing eventually constant 0 to win all games she can win.

To see that the energy version is not positionally determined, we consider the one-player game in Figure 3

Where the game starts at v, and v has the colour 1 and energy delta 1 and u has colour 0 and energy delta -1. Player 1 can win by playing according to the colours $101100111 \dots$, which keeps the energy non-negative, but requires infinite memory. However, any finite memory strategy winning the underlying game has to produce a colour sequence of the form $w0^\omega$, which will cause the energy to diverge to $-\infty$. ☐

To prove the existence of finite-memory winning strategies for Player 1 in the disjunctive case, we further assume that the potential winner after some history in the original game is decidable by a finite automaton.

Lemma 39. Let $g = \langle (V, E), v_0, V_1, V_2, W\rangle$ be a win/lose two-player game on a finite graph such that the simple restrictions of its product by finite automata are determined, and whenever Player 1 can win, she can do so *via* strategies using finite memory (resp. $m(n)$ bits of memory, where n is the number of states). Let a finite automaton $\mathcal{A}_g$ accept exactly the pairings of the histories that correspond to the future games won by Player 2, let $\mathcal{A}$ be a finite automaton over the alphabet E, and let us derive $g_{\mathcal{A}}$ from g by replacing W with $W_{\mathcal{A}}$ such that $\rho \in W_{\mathcal{A}}$ iff $\rho \in W \vee f(Pref(\rho)) \cap L_{\mathcal{A}} = \emptyset$. If Player 1 has a winning strategy in $g_{\mathcal{A}}$, he has one using finite (resp. $m(|V|\cdot|Q|\cdot|Q_g|) + \log|Q| + \log|Q_g|$-bits of) memory, where Q and Q_g are the states of $\mathcal{A}$ and $\mathcal{A}_g$, respectively.

Proof. Let $\mathcal{A} = (E, Q, q^0, F, \Delta)$, let $\mathcal{A}_g = (E, Q_g, q_g^0, F_g, \Delta_g)$, and let $g' := g \times \mathcal{A} \times \mathcal{A}_g$. Let $S \subseteq V \times Q \times Q_g$ be the vertices of g' from where Player 2 cannot force the run to reach $V \times F \times F_g$. Let us make a case disjunction: First case, (v_0, q^0, q_g^0) is not in S, so in g' Player 2 can reach $V \times F \times F_g$ and win, by playing in some way.

Second case, $(v_0, q^0, q_g^0) \in S$. If a finite history h in $g' \mid_S$ reaches $V \times F \times Q_g$, it must be in $V \times F \times (Q_g \setminus F_g)$, by definition of S. So Player 1 can win after h in g', and therefore in the simple restriction $g' \mid_S$ too. By assumption he can do so *via* a strategy using finite memory (resp. $m(|V|\cdot|Q|\cdot|Q_g|)$ bits of memory). He can use the same strategy to win $g_{\mathcal{A}}$, but he needs $\log|Q| + \log|Q_g|$ additional bits of memory to run $\mathcal{A}$ and $\mathcal{A}_g$. □

Since the assumptions from Lemma 39 are stronger than these from Lemma 34, we also find that Player 2 has a finite-memory winning strategy if she can win. However, even under the assumptions from Lemma 39 the memory required by Player 2 might not be uniformly bounded. We can obtain bounds for the memory required by Player 2 (Lemma 41) by considering subgame-perfect strategies. A strategy is subgame-perfect if it wins all future games that can be won by its player.

Lemma 40. Let Player 1 be able to win g and all its future games that he wins by some finite-memory strategy using k bits. Moreover, let for each strategy using k bits, let there be a finite automaton (using l bits) accepting exactly those histories won by that strategy. Then Player 1 has a finite-memory subgame-perfect strategy using $(l+k)(2^k|V|)^{2^{k|V|}} + 2^k|E|\log(2^k|E|)$ bits.

Proof. Omitted (see Section B in the appendix). □

Lemma 41. Let $g = \langle (V, E), v_0, V_1, V_2, W\rangle$ be a win/lose games on a finite graph such that Player 2 has a subgame-perfect strategy using m bits, and that there is an automaton deciding who wins from a history using l bits. Let $\mathcal{A}$ be a finite automaton over the alphabet E using k bits. Let us derive $g_{\mathcal{A}}$ from g by replacing W with $W_{\mathcal{A}}$ such that $\rho \in W_{\mathcal{A}}$ iff $\rho \in W \vee f(Pref(\rho)) \cap L_{\mathcal{A}} = \emptyset$. If Player 2 wins $g_{\mathcal{A}}$, she has a winning strategy using at most $m + l + k + 1$ bits of memory.

Proof. We can combine the automaton deciding who wins a history and $\mathcal{A}$ into one automaton using $l+k$ bits that accepts the intersection of these two languages. By considering the expansion by this automaton, we see that simulating it suffices for Player 2 to force an accepted history, if she can do so. Now reaching such a history and then switching to the subgame-perfect strategy using m bits wins $g_{\mathcal{A}}$ if this is possible at all, and requires $m + l + k + 1$ bits. □

We briefly return to the two versions of bounded energy conditions from (Bouyer et al. 2008): Either any energy gained in excess of the upper bound is just lost (as in e.g. recharging a battery), or gaining energy in excess of the bound leads to a loss of the protagonist (as in e.g. refilling a fuel tank without automatic spill-over prevention). Since both conditions can be simulated by a finite automaton, Lemma 37 implies the following.

Corollary 42. Battery-like energy parity games and spill-over-like energy parity games are determined *via* strategies using $\log E_{\max}$ bits of memory, where $E_{\max}$ is the energy bound in both cases.

Note that being able to win using bounded finite memory, and being able to decide who wins from a given history with a finite automaton together do not suffice to imply the existence of a subgame-perfect finite memory strategy.

Example 43. Consider a graph where edges are coloured by $0, 1, E, U$, where a vertex has an outgoing edge coloured by E also has one coloured by U and vice versa, and each such vertex is controlled by Player 1. Player 1 wins any game where there is some history of the form wE with $w \in \{0,1\}^*$ and w has the same number of 0 and 1s, and whenever there is a history of the form wU with $w \in \{0,1\}^*$ and w has different numbers of 0 and 1s. Now there is a finite automaton deciding which histories are won for Player 1, and linear memory suffices to win any such future game, but Player 1 has no finite-memory subgame-perfect strategy.

Proof. Player 1 wins from some not-yet-determined history iff he can force a vertex with E and U outgoing edges. Linear memory then suffices to keep track of how many 0 and 1's have been encountered along the way (as a simple path will suffice), and enables Player 1 to choose correctly. However, which choice is correct depends on the pre-history in a way that a finite automaton cannot keep track of. Thus, there is no finite-memory subgame-perfect strategy for Player 1. □

Note that the games in the example above are not closed by simple restriction.

Outlook

We expect the existence of some reasonable assumptions, akin to Lemma 41, that would lead to further generalizations of the results in this section. Especially, we should thus be able to prove that boolean combinations involving many regular languages and one original condition W preserve finite-memory determinacy. Such a result would in particular apply to multidimensional energy parity objectives.

Multidimensional objectives have recently received much attention, e.g. (Velner et al. 2015; Brenguier and Raskin 2015; Clemente and Raskin 2015). In the two-player win/lose version, there are multiple instances of the relevant data for a criterion (e.g. different type of energy, with each vertex having a cost of each type). In order to win, the protagonist has to satisfy all conditions at once. While multidimensional energy parity objectives are face the same obstacle as we had to resolve with Theorem 30 for the one-dimensional case, we can already consider multidimensional energy games and combine our Theorem 5 with the main result from (Velner et al. 2015) to obtain:

Corollary 44. Multi-player multi-outcome multi-dimensional energy games have Nash equilibria built from finite memory strategies.

Acknowledgments

We are grateful to Thomas Brihaye for pointing us to Julie De Pril's PhD thesis, and to Jean-François Raskin for suggesting the investi-

gation of energy parity games in this context. The investigation towards Theorem 5 was sparked by a question of Benjamin Aminof.

References

B. Aminof and S. Rubin. First cycle games. In *Proc. of Strategic Reasoning*, 2014. URL `http://www.arxiv.org/abs/1404.0414`.

P. Bouyer, U. Fahrenberg, K. G. Larsen, N. Markey, and J. Srba. Infinite runs in weighted timed automata with energy constraints. In F. Cassez and C. Jard, editors, *Formal Modeling and Analysis of Timed Systems*, volume 5215 of *Lecture Notes in Computer Science*, pages 33–47. Springer Berlin Heidelberg, 2008. ISBN 978-3-540-85777-8. doi: 10.1007/978-3-540-85778-5_4. URL `http://dx.doi.org/10.1007/978-3-540-85778-5_4`.

R. Brenguier and J.-F. Raskin. Pareto curves of multidimensional mean-payoff games. In *Computer Aided Verification*, volume 9207 of *Lecture Notes in Computer Science*, pages 251–267. Springer, 2015. ISBN 978-3-319-21667-6. doi: 10.1007/978-3-319-21668-3_15. URL `http://dx.doi.org/10.1007/978-3-319-21668-3_15`.

T. Brihaye, V. Bruyère, and J. De Pril. Equilibria in quantitative reachability games. In *Proc. of CSR*, volume 6072 of *LNCS*. Springer, 2010.

T. Brihaye, J. De Pril, and S. Schewe. Multiplayer cost games with simple Nash equilibria. In *Logical Foundations of Computer Science*, LNCS, pages 59–73, 2013. doi: 10.1007/978-3-642-35722-0_5. URL `http://dx.doi.org/10.1007/978-3-642-35722-0_5`.

T. Brihaye, G. Geeraerts, A. Haddad, and B. Monmege. To reach or not to reach? Efficient algorithms for total-payoff games. arXiv 1407.5030, 2014.

N. Bulling and V. Goranko. How to be both rich and happy: Combining quantitative and qualitative strategic reasoning about multi-player games (extended abstract). In *Proc. of Strategic Reasoning*, 2013. URL `http://www.arxiv.org/abs/1303.0789`.

A. Chakrabarti, L. de Alfaro, T. A. Henzinger, and M. Stoelinga. Resource interfaces. In R. Alur and I. Lee, editors, *Embedded Software*, volume 2855 of *Lecture Notes in Computer Science*, pages 117–133. Springer Berlin Heidelberg, 2003. ISBN 978-3-540-20223-3. doi: 10.1007/978-3-540-45212-6_9. URL `http://dx.doi.org/10.1007/978-3-540-45212-6_9`.

K. Chatterjee and L. Doyen. Energy parity games. *Theor. Comput. Sci.*, 458:49–60, 2012. doi: http://dx.doi.org/10.1016/j.tcs.2012.07.038.

L. Clemente and J. Raskin. Multidimensional beyond worst-case and almost-sure problems for mean-payoff objectives. In *30th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2015, Kyoto, Japan, July 6-10, 2015*, pages 257–268, 2015. doi: 10.1109/LICS.2015.33. URL `http://dx.doi.org/10.1109/LICS.2015.33`.

J. Fearnley, D. Peled, and S. Schewe. Synthesis of succinct systems. In S. Chakraborty and M. Mukund, editors, *Automated Technology for Verification and Analysis*, Lecture Notes in Computer Science, pages 208–222. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-33385-9. doi: 10.1007/978-3-642-33386-6_18. URL `http://dx.doi.org/10.1007/978-3-642-33386-6_18`.

N. Fijalkow and M. Zimmermann. Cost-parity and cost-streett games. In T. K. D. D'Souza and J. Radhakrishan, editors, *32nd Int. Conf. on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 18 of *LIPIcs*, 2012.

N. Fijalkow and M. Zimmermann. Parity and streett games with costs. *Logical Methods in Computer Science*, 10(2), 2014.

Y. Gurevich and L. Harrington. Trees, automata and games. In *Proc. STOC*, 1982.

M. Jurdzinski, M. Paterson, and U. Zwick. A deterministic subexponential algorithm for solving parity games. *SIAM J. Comput.*, 38(4):1519–1532, 2008. doi: 10.1137/070686652.

S. Le Roux. Infinite sequential Nash equilibria. *Logical Methods in Computer Science*, 9(2), 2013.

S. Le Roux and A. Pauly. Infinite sequential games with real-valued payoffs. In *CSL-LICS '14*, pages 62:1–62:10. ACM, 2014a. doi: 10.1145/2603088.2603120. URL `http://doi.acm.org/10.1145/2603088.2603120`.

S. Le Roux and A. Pauly. Infinite sequential games with real-valued payoffs. arXiv:1401.3325, 2014b. URL `http://arxiv.org/abs/1401.3325`.

S. Le Roux and A. Pauly. Weihrauch degrees of finding equilibria in sequential games. arXiv:1407.5587, 2014c.

S. Le Roux and A. Pauly. Weihrauch degrees of finding equilibria in sequential games. In A. Beckmann, V. Mitrana, and M. Soskova, editors, *Evolving Computability*, volume 9136 of *Lecture Notes in Computer Science*, pages 246–257. Springer, 2015. doi: 10.1007/978-3-319-20028-6_25. URL `http://dx.doi.org/10.1007/978-3-319-20028-6_25`.

J. F. Mertens. Repeated games. In *Proc. Internat. Congress Mathematicians*, pages 1528–1577. American Mathematical Society, 1987.

S. Paul and S. Simon. Nash Equilibrium in Generalised Muller Games. In R. Kannan and K. N. Kumar, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 4 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 335–346, Dagstuhl, Germany, 2009. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi: http://dx.doi.org/10.4230/LIPIcs.FSTTCS.2009.2330. URL `http://drops.dagstuhl.de/opus/volltexte/2009/2330`.

J. D. Pril. *Equilibria in Multiplayer Cost Games*. PhD thesis, Université de Mons, 2013.

Y. Velner, K. Chatterjee, L. Doyen, T. A. Henzinger, A. Rabinovich, and J.-F. Raskin. The complexity of multi-mean-payoff and multi-energy games. *Information and Computation*, 241:177 – 196, 2015. ISSN 0890-5401. doi: http://dx.doi.org/10.1016/j.ic.2015.03.001. URL `http://www.sciencedirect.com/science/article/pii/S0890540115000164`.

A. M-reachability games

In this section let Σ be a finite alphabet of cardinality n.

A.1 Proof of Theorem 18

Definition 45.
- For $a \in \Sigma$ and $w, w' \in \Sigma^*$, let $a \in w$ mean that a occurs in w, and $w \subseteq w'$ that the letters in w occur in w', and $w \sim w'$ that $w \subseteq w'$ and $w' \subseteq w$.
- For all $a \in \Sigma$ and $u, v, w \in \Sigma^*$ such that $v \subseteq ua$, the word uaw is a forward loop-cut of $uavaw$. Let $\to_f$ be the transitive closure of this relation, in particular $uavaw \to_f uaw$.
- For all trees T over Σ, let $[T]_M$ be the set of the maximal paths of T, from the root to the leaves if T is finite.
- Let T_0, T_1, T_2 be finite trees over Σ, let $a \in \Sigma$ and $w, w' \in \Sigma^*$ be such that $wa \in [T_0]_M$ and $w'a \in [T_1]_M$ and $w' \subseteq wa$. Then the tree $T_0 \cup waT_2$ is a forward loop-cut of the tree $T_0 \cup wa(T_1 \cup w'aT_2)$. Let $\to_f$ be the transitive closure of this relation, in particular $T_0 \cup wa(T_1 \cup w'aT_2) \to_f T_0 \cup waT_2$.

Lemma 46.
1. For all $u, v, v', w \in \Sigma^*$ if $v \to_f v'$ then $uvw \to_f uv'w$.
2. Let $w \in \Sigma^*$ be terminal for $\to_f$. Then $|w| \leq \frac{n(n+1)}{2}$, and the bound is tight.
3. On linear trees, $\to_f$ for trees coincides (up to isomorphism) with $\to_f$ for words.
4. $\to_f$ for trees terminates.
5. Let T be a finite tree over Σ such that some word in $[T]_M$ is non-terminal for $\to_f$. Then T is also non-terminal for $\to_f$.
6. Let T be a finite tree over Σ. If T is terminal for $\to_f$, its height is at most $\frac{n(n+1)}{2}$.
7. Let T be a finite tree over Σ and let $T \to_f T'$. Then for all $w' \in [T']_M$ there exists $w \in [T]_M$ such that $w \sim w'$.
8. If t is a prefix of a tree T over Σ, and if $t \to_f t'$, there exists an extension T' of t' such that $T \to_f T'$.

Proof.
1. Straightforward.
2. By induction on n. The claim holds for $n = 1$, so let us assume that $n > 1$. Let us decompose w into uau' where $a \notin u$ and $u' \subseteq ua$, so $|u'| \leq n - 1$. By Lemma 46.1 and induction hypothesis $|u| \leq \frac{n(n-1)}{2}$, so $|uau'| \leq \frac{n(n-1)}{2} + 1 + n - 1 = \frac{n(n+1)}{2}$. For the tightness of the bound let $\Sigma = \{1, 2, \dots, n\}$ and note that the word $1213124 \dots k123 \dots (k-1)k+1 \dots n123 \dots n-1$ is terminal for $\to_f$ and has length $\frac{n(n+1)}{2}$.
3. Straightforward.
4. The number of vertices decreases.
5. Let $waw'aw'' \in [T]_M$ be such that $w' \subseteq wa$. Let us decompose T as $T_0 \cup wa(T_1 \cup w'aT_2)$ where $T_0 := \{u \in T \mid \neg(wa \sqsubset u)\}$ and $T_1 := \{u \in \Sigma^* \mid waw' \in T \wedge \neg(waw'a \sqsubset u)\}$ and $T_2 := \{u \in \Sigma^* \mid waw'au \in T\}$. Then $T \to_f T_0 \cup waT_2$.
6. By combining Lemmas 46.2 and 46.5
7. Let $T \to_f T'$ be witnessed by T_0, T_1, T_2, a, w, w', and let $u \in [T']_M$. Let us make a case disjunction. If u is not a proper extension of wa, then $u \in [T]_M$. If $u = wav$ for some v (in $[T_2]_M$), $waw'av \in [T]_M$ and $wav \sim waw'av$ since $w' \subseteq wa$.
8. Straightforward.

$\square$

Proof of Theorem 18. Let us first prove that if a player has a strategy inducing a tree T (subtree of the unraveling of the game) such that $T \to_f T'$ (even without the $w' \subseteq wa$ condition), the player can also induce T'. Let T_0, T_1, T_2 be trees over Σ, let $a \in \Sigma$ and $w, w' \in \Sigma^*$ be such that $wa \in [T_0]_M$ and $w'a \in [T_1]_M$. Let us assume that Player 1 can induce $T := T_0 \cup wa(T_1 \cup w'aT_2)$ from the starting vertex of the game. This shows that starting from the vertex a she can induce the tree $T_1 \cup w'aT_2$, she does it after history wa, and also the tree T_2, she does it after history $waw'a$. But the ability to induce one tree starting from one vertex does not depend on the history that lead there, so she could also induce T_2 after history wa, thus overall inducing $T_0 \cup waT_2$.

Let s be a winning strategy for Player 1. It induces an infinite subtree T_s such that each infinite path has a (smallest) finite prefix that witnesses victory. Let us cut the rest of the tree after these smallest prefixes. By König's Lemma the resulting tree $F(T_s)$ is finite. If $F(T_s) \to_f T'$ then by the claim we have just proved above Player 1 has a strategy s' such that $T' = F(T_{s'})$. Let us $\to_f$-reduce $F(T_s)$ until some$\to_f$-terminal t'. By the claim above and Lemma 46.8, Player 1 has a strategy s' such that $F(T_{s'}) = t'$. By Lemma 46.7 the strategy s' is winning, and by Lemma 46.6 it is doing so in at most $\frac{n(n+1)}{2}$ steps. $\square$

A.2 Towards bound improvement

This section shows that the reduction $\to_f$ and the bounds may be improved if one is only concerned about preserving the starting, visited, and ending vertices in words. Unfortunately a natural variant of Lemma 46.7 is more difficult to prove, or even false.

Definition 47.
- For a set S and a word w let $S \subseteq w$ mean that all elements in S occurs in w and $S \cap w = \emptyset$ mean that no element in S occurs in w.
- For all $a \in \Sigma$ and $u, v, w \in \Sigma^*$ the word uaw is a loop-cut of $uavaw$. Let $\sqsubseteq_{lc}$ be the transitive closure of this. *E.g*, $uaw \sqsubseteq_{lc} uavaw$.
- For all $m \leq n$ let $lc(m, n) := \sup_{w \in \Sigma^* \wedge S \subseteq w \wedge |S| = m} min_{v \sqsubseteq_{lc} w, S \subseteq v} |v|$.
- Let T, T', T'' be trees over Σ, let $a \in \Sigma$ and $w, w' \in \Sigma^*$ be such that $wa \in [T]_M$ and $w'a \in [T']_M$. Then $T \cup waT''$ is a loop-cut of $T \cup wa(T' \cup w'aT'')$ along $(wa, w'a)$. Let $\sqsubseteq_{lc}$ be the transitive closure of this.

When loop-cutting a word over Σ with the objective of preserving a subset of them of size m, a natural question is whether we are likely to reach a short word. $lc(m, n)$ is the guarantee how short it gets. Note that the loop-cut preserves the first and last letters of a word.

Lemma 48. $lc(m, n) \leq (m + 1)(n - \frac{m}{2})$.

Proof. By induction on m, which holds for $m = 0, 1$. So let $m > 0$, let $w \in \Sigma^*$, and let $S \subseteq w$ be such that $|S| = m$. So w can be decomposed as uav where $u \cap S = \emptyset$ and $a \in S$. Since u involves only $n - m$ different letters, it can be reduced by loop-cut to u' such that $|u'| \leq n - m$. Furthermore, by induction hypothesis av can be reduced by loop-cut to av' such that $S \setminus \{a\} \subseteq av'$ and $|av'| \leq lc(m - 1, n) \leq m(n - \frac{m-1}{2})$. So $u'av' \sqsubseteq_{lc} w$ and $S \subseteq u'av'$ and $|u'av'| \leq n - m + m(n - \frac{m-1}{2}) = (m+1)(n - \frac{m}{2})$. $\square$

Corollary 49. $\frac{n^2}{4} \leq lc(n, n) \leq \frac{n(n+1)}{2}$

Proof. The upper bound comes from Lemma 48. For the lower bound, let $k \in \mathbb{N}$ and let G_k be the graph with vertices $x_1, \dots, x_k$ and $y_1, \dots, y_k$ with the arcs (x_i, x_{i+1}) for all possible i, and (x_k, y_i) and (y_i, x_1) for all i. Every path in G_k starting at x_1, visiting all the vertices, and ending in x_k has length at least $(k+1)^2$. Setting $n = 2k$ gives the bound $(\frac{n}{2} + 1)^2$. The case where n is odd is similar. $\square$

Lemma 50. $lc(n, n) \leq \frac{n^2}{4} + n + 1$ and equality holds when n is even.

$$\begin{array}{ccccccccc} (2\ldots n) & 123\ldots k & (1\ldots k-1) & k+1 & (1\ldots k) & \ldots n-1 & (1\ldots n-2) & n & (1\ldots n-1) \\ \leq k-2 & & \leq k-2 & & \leq k-2 & & \leq k-2 & & \leq k-1 \end{array}$$

Figure 4

$$\begin{array}{ccccccccccc} (1\ldots n-1) & n\ldots & (1\ldots m) & m+1 & (2\ldots m) & 123\ldots k & (1\ldots k-1) & k+1\ldots & m & (1\ldots m-1) \\ \leq k-2 & & \leq k-3 & & \leq k-3 & & \leq k-2 & & & \leq k-1 \end{array}$$

Figure 5

Proof. Let w be a word over $\{1, 2, \ldots, n\}$ such that no loop-cut of w involves all the n numbers. Up to renaming let $123\ldots k$ be the leftmost longest repetition-free factor of w, and let $k = |v|$. Let us make a case disjunction. First case, every letter occurring before $123\ldots k$ also occurs in $123\ldots k$ or after. So the word can be decomposed as in Figure 4 up to renaming.

For example $(2\ldots n)$ above represents repetiton-free words involving only numbers between 2 and n. The lengths of these type of words are bounded below them. The overall length is then less than or equal to $n + 1 + (k - 2)(n - k + 2)$ which is (upper) bounded by $\frac{n^2}{4} + n + 1$. (It is similar if $123\ldots k$ occurs near the leftmost end of w.)

Second case, some numbers occur only before $123\ldots k$, and some only after it, so w can be decomposed as in Figure 5, first towards the right, and then towards the left.

The overall length is then less than or equal to $n + 1 + (k - 2)(m - k + 2) + k - 1 + (k - 3(n - m - 1)) = m = n+1+(k-2)(n-k+1)+n-m$ which is also (upper) bounded by $\frac{n^2}{4} + n + 1$.

□

B. Other omitted proofs

Lemma (Lemma 22)**.** For a winning strategy s of Player 1 in g exactly one of the following holds:

1. s is also winning in g'.
2. There is a bad history $h \in \mathcal{H}(s)$.

Proof. $\neg 1. \Rightarrow 2.$ Let Player 2 have a strategy t that is winning g' against s. As by assumption t loses against s in g, and the Muller conditions are identical in g and g', we see that in g' the energy condition for Player 1 must be violated for that play. This happens at some history h'. As energy levels in g and g' can only differ due to losses causes by the energy cap, h' must have some prefix h that in g would have had an energy level exceeding $\frac{n(n+1)}{2}W$. W.l.o.g. we may assume h to be locally maximal, and the shortest prefix of h' with this property. This in particular implies that h has energy level $\frac{n(n+1)}{2}W$ in g', so the energy difference between h and h' in g' exceeds $\frac{n(n+1)}{2}W$. This is a lower bound for the energy difference between h and h' in g (as the only difference could be further losses due to the energy cap), and thus h' witnesses that h is a bad history.

$2. \Rightarrow \neg 1.$ Let $h' \in B$ witness that h is bad. Player 2 has a strategy to reach h' against s, and we claim that this strategy will win against s in g'. As the energy level in g' is capped at $\frac{n(n+1)}{2}W$, the energy level at h in g' will be at most $\frac{n(n+1)}{2}W$. But as the energy difference between h and h' is more than $\frac{n(n+1)}{2}W$, this implies that in g' the energy level at h' will be negative, thus Player 1 loses.

□

Lemma (Lemma 40)**.** Let Player 1 be able to win g and all its future games that he wins by some finite-memory strategy using k bits. Moreover, let for each strategy using k bits, let there be a finite automaton (using l bits) accepting exactly those histories won by that strategy. Then Player 1 has a finite-memory subgame-perfect strategy using $(l + k)(2^k|V|)^{2^k|V|} + 2^k|E|\log(2^k|E|)$ bits.

Proof. There are $(2^k|V|)^{2^k|V|}$ strategies using k bits. For each of them we keep track of its current memory content (using k bits), and run the automaton deciding whether it is currently winning (using l bits). We always play according to some strategy (which we store using $2^k|E|\log(2^k|E|)$ bits), and we change it only once it is no longer winning, and another strategy is identified as winning from the current history. □

Lemma (Lemma 32)**.** The valuation-preference combinations in MMEP games are automatic-piecewise prefix-linear.

Proof. The pieces are defined by the current and least energy levels, i.e. the values E^a_n and $\min_{j\leq n} E^a_n$. As energy is bounded, both can easily be computed by a finite automaton. If h and h' end with the same vertex and share the same current energy level and least energy level, then the least energy level reached in hp is equal to the least energy level reached in $h'p$. As the least priority seen infinitely many times depends only on the tail, but never on the finite prefix, we see that for h and h' being the same piece, hp and hp' are interchangeable for the player. □