Information and Computation 254 (2017) 195–216

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

First-cycle games ☆

Benjamin Aminof [a], Sasha Rubin [b,*,1]

[a] Technische Universität Wien, Austria
[b] Università degli Studi di Napoli "Federico II", Italy

ARTICLE INFO

Article history:
Received 21 November 2014
Available online 4 November 2016

Keywords:
Graph games
Cycle games
Memoryless determinacy
Parity games
Mean-payoff games
Energy games

ABSTRACT

First-cycle games (FCG) are played on a finite graph by two players who push a token along the edges until a vertex is repeated, and a simple cycle is formed. The winner is determined by some fixed property Y of the sequence of labels of the edges (or nodes) forming this cycle. These games are intimately connected with classic infinite-duration games such as parity and mean-payoff games. We initiate the study of FCGs in their own right, as well as formalise and investigate the connection between FCGs and certain infinite-duration games.

We establish that (for efficiently computable Y) the problem of solving FCGs is Pspace-complete; we show that the memory required to win FCGs is, in general, $\Theta(n)!$ (where n is the number of nodes in the graph); and we give a full characterisation of those properties Y for which all FCGs are memoryless determined.

We formalise the connection between FCGs and certain infinite-duration games and prove that strategies transfer between them. Using the machinery of FCGs, we provide a recipe that can be used to very easily deduce that many infinite-duration games, e.g., mean-payoff, parity, and energy games, are memoryless determined.

1. Introduction

Infinite-duration games are studied in computer science in the context of decidability of logical theories as well as verification and synthesis in formal methods. In particular, reactive systems, which consist of an ongoing interaction between a program and its environment, may be formalised as a game played on a graph, by two players, who push a token along the edges of the graph, resulting in an infinite path, called a play. The winner of the play is determined by some winning condition. For example, in (one version of) mean-payoff games each edge in the graph carries a real number, called a weight, and Player 0 wins the play if and only if the limit-supremum of the running averages of the weights is positive; and otherwise Player 1 wins. Other types of games from the formal verification literature are reachability games, Büchi games, parity games, Muller games, energy games, etc.

Intuitively, certain games on finite graphs can be won using greedy reasoning. For instance, to win a mean-payoff game, it is sufficient for Player 0 to ensure that the average weight of each cycle that is formed is positive. This, in turn, can be ensured by enforcing the average weight of the *first cycle* formed to be positive (because Player 0 could then "forget" that the cycle was formed, and play another cycle with positive average weight, and so on). Thus, in some cases, one can

☆ Some of the results in this paper were reported in the Proceedings of the Second International Workshop on Strategic Reasoning (SR), April 2014.

* Corresponding author.
E-mail addresses: benj@forsyte.at (B. Aminof), rubin@unina.it (S. Rubin).

[1] Marie Curie fellow of the Istituto Nazionale di Alta Matematica.

http://dx.doi.org/10.1016/j.ic.2016.10.008
0890-5401/

reduce reasoning about infinite-duration games to *first-cycle games*, i.e., games in which the winner is determined by the first cycle on the play. Such reasoning appears in [6], where first-cycle mean-payoff games were defined and used to prove that mean-payoff games can be won using memoryless strategies (i.e., the next move of a player does not depend on the full history up till now, but only on the current node of the play). Memoryless strategies are extremely useful, e.g., they are used to prove deep results in the theory of automata (e.g., Rabin's theorem that automata on infinite trees can be complemented), and to prove upper bounds on the complexity of solving certain classes of games (e.g., that solving parity games is in NP $\cap$ co-NP).

In this paper we study first-cycle games in their own right and formalise the greedy reasoning above which connects first-cycle games with certain infinite-duration games. We now discuss our main contributions.

First-cycle games. We define first-cycle games (FCGs). These games are played on a finite graph (called an arena) by two players who push a token along the edges of the graph until a cycle is formed. Player 0 wins the play if the sequence of labels of the edges (or nodes) of the cycle is in some fixed set Y, and otherwise Player 1 wins. The set Y is called a cycle property, and we say that a cycle satisfies Y if its sequence of labels is in Y. For example, if every vertex is labelled by an integer priority, and $Y = \mathsf{cyc\text{-}Parity}$ comprises sequences whose largest priority is even, then a cycle satisfies cyc-Parity if the largest priority occurring on the cycle is even. For a fixed cycle property Y, we write $\textsc{Fcg}(Y)$ for the family of games over all possible arenas with this winning condition.

Complexity and memory requirements. We give a simple example showing that first cycle games (FCGs) are not necessarily memoryless determined. We then show that, for a graph with n nodes, whereas no winning strategy needs more than $n!$ memory (since this is enough to remember the whole history of the game), some winning strategies require at least $(\frac{n-1}{3})!$ memory (Proposition 1, Page 201). We analyse the complexity of solving FCGs and show that it is Pspace-complete. More specifically, we show that if one can decide in Pspace whether a given cycle satisfies the property Y, then solving the games in $\textsc{Fcg}(Y)$ is in Pspace; and that there is a trivially computable cycle property Y for which solving the games in $\textsc{Fcg}(Y)$ is Pspace-hard (Theorem 1, Page 202).

First-cycle games and infinite-duration games (Section 5). The central object that connects cycle games with infinite-duration games is the *cycles-decomposition* of a path (used for example by [12] to derive the value of a mean-payoff game). Informally, a path is decomposed by pushing the edges of the path onto a stack and, as soon as a cycle is detected in the stack, the cycle is output, popped, and the algorithm continues. This decomposes all but finitely many edges of the path into a sequence of simple cycles.

In order to connect FCGs with infinite-duration games we define the following notion: a winning condition W (such as the parity winning condition) is *Y-greedy on arena $\mathcal{A}$* if, in the game on arena $\mathcal{A}$ with winning condition W, Player 0 is guaranteed to win by ensuring that every cycle in the cycles-decomposition of the play satisfies Y, and Player 1 is guaranteed to win by ensuring that every cycle in the cycles-decomposition does not satisfy Y (Definition 5, Page 208). We prove a *Strategy Transfer Theorem:* if W is Y-greedy on $\mathcal{A}$ then the winning regions in the following two games on arena $\mathcal{A}$ coincide, and memoryless winning strategies transfer between them: the infinite-duration game with winning condition W, and the FCG with cycle property Y (Theorem 7, Page 209).

To illustrate the usefulness of being Y-greedy, we instantiate the definition to well-studied infinite-duration games such as parity, mean-payoff, and energy games.

Memoryless determinacy of first-cycle games. We next address the fundamental question: for which cycle properties Y is every game in $\textsc{Fcg}(Y)$ memoryless determined (i.e., no matter the arena)? We provide sufficient and necessary conditions for all games in $\textsc{Fcg}(Y)$ to be memoryless determined (Theorem 6, Page 208). Although applying this characterisation may not be hard, it involves reasoning about arenas, which is sometimes inconvenient. Therefore, we also provide the following easy-to-check conditions on Y that ensure memoryless determinacy for all games in $\textsc{Fcg}(Y)$ (Theorem 9, Page 210): Y is closed under cyclic permutations (i.e., if $ab \in Y$ then $ba \in Y$), and both Y and its complement are closed under concatenation (a set of strings X is closed under concatenation if $a, b \in X$ implies $ab \in X$). We demonstrate the usefulness of these conditions by observing that natural cycle properties are easily seen to satisfy them, e.g., cyc-Parity, $\mathsf{cyc\text{-}MeanPayoff}_\nu$ (which states that the limsup average of the weights is at most ν), and cyc-Energy (which states that the sum of the weights is positive).

Easy-to-follow recipe. We integrate the previous results by providing an easy-to-follow recipe that allows one to deduce memoryless determinacy of all classic games that are memoryless determined and many others besides (Section 8). The recipe states that, given a winning condition W, first "finitise" W to get a cycle property Y that is closed under cyclic permutations, then prove that W is Y-greedy on the arenas of interest (usually all arenas), and finally, either show that both Y and its complement are closed under concatenation, or show that W is prefix-independent. For example, if W is the parity winning condition (i.e., the largest priority occurring infinitely often is even), the natural "finitisation" of W is the cycle property $Y = \mathsf{cyc\text{-}Parity}$ (i.e., sequences of priorities whose largest priority is even), and it is almost completely trivial to apply the recipe in this case.

Related work. The relationship of our work with that in [6] is as follows. First, our paper deals with qualitative games (i.e., a play is either won or lost) whereas [6] consider quantitative (i.e., a play is assigned a real number, which Player 0 wants to minimize and Player 1 wants to maximize) mean-payoff games. Their main result states that mean-payoff games

are memoryless determined. However, they simultaneously prove that first-cycle games with $Y = \text{cyc-MeanPayoff}_v$ are memoryless determined. We generalise (in the qualitative setting) both of these facts and their proofs.

Referring to their proofs, [6, Page 111] state: "Our proofs are roundabout, we use the infinite game F to establish facts about the finite game G and vice versa. Perhaps more direct proofs would be desirable." In contrast, the proof of the characterisation of memoryless determined FCGs (Theorems 4, 5 and 6) is direct, and does not go through infinite-duration games. Nonetheless, we use their idea of inducting on the choice nodes of the players, and of employing "reset" nodes in the arena.

We briefly discuss other related work. We point out (in Theorem 3, Page 204) that first-cycle games are not necessarily memoryless determined, even if the cycle property Y is closed under cyclic permutations contrary to the claim in [4, Page 370]. Our work thus also supplies a correct proof Lemma 4 in [5] which relied on this incorrect claim (see Remark 2, Page 210). Conditions that ensure (or characterise) which winning conditions always admit memoryless strategies appear in [3,8,10]. However, none of these exploit the connection to first-cycle games. For example, [8] give a full characterization of winning conditions for which all infinite-duration games are memoryless determined. Unlike our framework, the main objects of study in their work are winning conditions (i.e., languages of infinite sequences of labels) and one cannot use their results to reason about cycles in an arena since every cycle in an arena induces a cycle in the sequences of labels, but not vice versa. Their characterisation of those winning conditions which admit memoryless determined games is very "infinite" in nature, as it involves reasoning about preference relations among infinite words, and their properties concerning all infinite subsets of words recognizable by nondeterministic automata. On the other hand, their result, that if all solitaire games with a given winning condition are memoryless determined then so are all the two player games, provides a much more useful tool.

The present paper differs from the preliminary version of this work [1] mainly in the following respects: we have re-organised some of the definitions, supplied all proofs, established a full characterisation of those cycle properties Y such that every FCG over Y and Y-greedy games are memoryless determined, and extended the easy-to-follow recipe to include the case that the winning condition is prefix-independent.

2. Games

In this paper all games are two-player turn-based games of perfect information played on finite graphs. The players are called Player 0 and Player 1.

Arenas. An *arena* is a labelled directed graph $\mathcal{A} = (V_0, V_1, E, \mathbb{U}, \lambda)$ where

1. V_0 and V_1 are disjoint sets of *vertices* (alternatively, *nodes*) of Player 0 and Player 1, respectively; the set of vertices of the arena is $V := V_0 \cup V_1$, and is assumed to be finite and non-empty.
2. $E \subseteq V \times V$ is a set of *edges* with no dead-ends (i.e., for every $v \in V$ there is some edge $(v, w) \in E$).
3. $\mathbb{U}$ is a set of possible *labels* (typical choices for $\mathbb{U}$ are $\mathbb{Q}$ and $\mathbb{N}$).
4. $\lambda : E \to \mathbb{U}$ is a *labelling function* (which will be used by the winning condition).

A *sub-arena* of $\mathcal{A}$ is an arena of the form $(V_0, V_1, E', \mathbb{U}, \lambda')$ where $E' \subseteq E$ and $\lambda'(e) = \lambda(e)$ for $e \in E'$, i.e., it may have fewer edges. If $V_0 = \emptyset$ or $V_1 = \emptyset$ then $\mathcal{A}$ is called a *solitaire arena*.

Remark 1. Note that all the results in this paper also hold for vertex labelled arenas, by labelling every edge (v, w) by the label of its source v. In particular, we use vertex labelling in some of our examples, with this transformation in mind. Also note that transforming a vertex labelled arena to an edge-labelled one can be done by inserting an intermediate node in the middle of every edge, and giving it the edge label. However, since this transformation changes the structure of the arena, some properties that hold for vertex labelled arenas do not always transfer to edge-labelled ones.

Sequences. Let $\mathbb{N}$ denote the positive integers. The ith element (for $i \in \mathbb{N}$) in a sequence u is denoted u_i. The *length* of u, denoted $|u|$, is the cardinality of the sequence u. For $1 \le i \le j \le |u|$, write $u[i, j]$ for the sub-sequence $u_i u_{i+1} \cdots u_j$. The empty sequence is denoted ϵ. By convention, $u[i, j] = \epsilon$ if $j < i$. The set of infinite sequences over alphabet X is written X^ω, the set of finite sequences is written X^*. We write concatenation as $u \cdot v$ or simply as uv.

Paths, simple paths, and node-paths. For an edge $e = (v, w)$ we call v the *start* and w the *end* of e, and write $\mathsf{start}(e) := v$ and $\mathsf{end}(e) := w$; we say that v and w *appear on the edge* e. A *path* π in $\mathcal{A}$ is a finite or infinite sequence of edges $\pi_1 \pi_2 \cdots \in E^* \cup E^\omega$ such that $\mathsf{end}(\pi_i) = \mathsf{start}(\pi_{i+1})$ for $1 \le i < |\pi|$. The set of paths in $\mathcal{A}$ is denoted $paths(\mathcal{A})$. We extend start and end to paths in the natural way: $\mathsf{start}(\pi) := \mathsf{start}(\pi_1)$, and if π is of length $\ell \in \mathbb{N}$ then $\mathsf{end}(\pi) := \mathsf{end}(\pi_\ell)$. We say that vertex v *appears on the path* π if v appears on some edge π_i of π. We extend λ from edges to paths point-wise: $\lambda(\pi)$ is defined to be the string of labels $\lambda(\pi_1)\lambda(\pi_2)\cdots$. A path π is called *simple* if $\mathsf{start}(\pi_i) = \mathsf{end}(\pi_j)$ implies $i = j + 1$.

A sequence of nodes $v \in V^* \cup V^\omega$ is called a *node-path* if $(v_i, v_{i+1}) \in E$ for all i. If $\pi \in E^* \cup E^\omega$ is a path then $\mathsf{nodes}(\pi)$ is a node-path, where $\mathsf{nodes}(\pi)$ is defined to be the sequence $\mathsf{start}(\pi_1)\mathsf{start}(\pi_2)\cdots$ if π is infinite, and $\mathsf{start}(\pi_1)\cdots\mathsf{start}(\pi_{|\pi|})\mathsf{end}(\pi_{|\pi|})$ if it is finite. Every node-path is either a singleton sequence v or of the form $\mathsf{nodes}(\pi)$

for some path π. For a finite non-empty sequence u of vertices (such as a finite node-path), we overload notation and write $\mathsf{end}(u)$ for the last node in u.

Histories and strategies. A strategy for a player is a function that tells the player what node to move to given the history, i.e., the sequence of nodes visited so far. Define the set $H_\sigma(\mathcal{A})$ of *histories* for Player $\sigma \in \{0,1\}$ by $H_\sigma(\mathcal{A}) := \{u \in V^*V_\sigma : (u_n, u_{n+1}) \in E, 1 \le n < |u|\}$. In other words, $H_\sigma(\mathcal{A})$ is the set of node-paths ending in V_σ. A *strategy for Player* σ is a function $S : H_\sigma(\mathcal{A}) \to V$ such that $(\mathsf{end}(u), S(u)) \in E$ for all $u \in H_\sigma(\mathcal{A})$. A strategy S for Player σ is *memoryless* if $S(u) = S(u')$ for all $u, u' \in H_\sigma(\mathcal{A})$ with $\mathsf{end}(u) = \mathsf{end}(u')$. Hence, we usually consider a memoryless strategy as a function $S : V_\sigma \to V$. A node-path $u \in V^* \cup V^\omega$ is *consistent* with a strategy S for Player σ if whenever $u_n \in V_\sigma$ (for $n < |u|$) then $u_{n+1} = S(u[1,n])$. A path π is *consistent* with a strategy S if $\mathsf{nodes}(\pi)$ is consistent with S.

A strategy S for Player σ is *generated by a Mealy machine* $\langle M, m_I, \delta, \mu\rangle$ if there exists a finite set M of *memory states*, an *initial state* $m_I \in M$, a *memory update* function $\delta : V \times M \to M$, and a *next-move function* $\mu : V_\sigma \times M \to V$, such that for a history $u = u_1u_2\cdots u_l \in H_\sigma(\mathcal{A})$ we have $S(u) = \mu(u_l, m_l)$ where m_l is defined inductively by $m_1 = m_I$ and $m_{i+1} = \delta(u_i, m_i)$. A strategy S is *finite-memory* if it is generated by some Mealy machine. A strategy S *uses memory at most* k if it is generated by some Mealy machine with $|M| \le k$, and it *uses memory at least* k if every Mealy machine generating S has $|M| \ge k$.

Notational abuse. We sometimes, for a path $\pi \in E^*$, write $S(\pi)$ instead of $S(\mathsf{nodes}(\pi))$.

Plays. An infinite path in $\mathcal{A}$ is called a *play*.[2] We denote the set of all plays of $\mathcal{A}$ by $plays(\mathcal{A})$. For a node $v \in V$, and strategy S, we write $plays_{\mathcal{A}}(S, v)$ for the set of plays π that start with v and are consistent with S; we write $reach_{\mathcal{A}}(S, v)$ for the set of vertices in V that appear on the plays in $plays_{\mathcal{A}}(S, v)$. We may drop the subscript $\mathcal{A}$ when the arena is clear from the context.

Games and winning. A *game* is a pair $(\mathcal{A}, O)$ consisting of an arena $\mathcal{A} = (V_0, V_1, E, \mathbb{U}, \lambda)$ and an *objective* $O \subseteq plays(\mathcal{A})$. Objectives are usually determined by winning conditions or cycle properties (defined later). A play π in a game $(\mathcal{A}, O)$ is *won by Player* 0 if $\pi \in O$, and otherwise it is *won by Player* 1. A strategy S for Player σ is *winning from* a node $v \in V$ (in the game $(\mathcal{A}, O)$) if every play in $plays_{\mathcal{A}}(S, v)$ is won by Player σ. If Player σ has a winning strategy from v we say that Player σ *wins from* v. A game $(\mathcal{A}, O)$ is said to be *determined* if, for every $v \in V$, one of the players wins from v.

The *winning region (resp. memoryless winning region)* of Player σ is the set of vertices $v \in V$ such that Player σ has a winning strategy (resp. memoryless winning strategy) from v. We denote the winning region in the game $(\mathcal{A}, O)$ of Player σ by $\mathsf{WR}^\sigma(\mathcal{A}, O)$.

Solitaire games and memoryless strategies. If either V_0 or V_1 is empty, then the game $(\mathcal{A}, O)$ is called a *solitaire game*.

A simple property of memoryless strategies that is often used is the following. In a game over an arena $\mathcal{A} = (V_0, V_1, E, \mathbb{U}, \lambda)$, every memoryless strategy S for Player σ induces a sub-arena $\mathcal{A}^{|S} = (V_0, V_1, E^{|S}, \mathbb{U}, \lambda^{|S})$, obtained by deleting all edges $(v, w) \in E$ where $v \in V_\sigma$ and $w \neq S(v)$. Observe that in $\mathcal{A}^{|S}$ all Player σ nodes have exactly one outgoing edge (namely the edge specified by S), and thus Player σ has no choices to make in this arena. For this reason, many times one considers the solitaire arena $\mathcal{A}^{\|S}$ in which all the nodes are assigned to Player $1-\sigma$, i.e., $\mathcal{A}^{\|S} = (V_0^{\|S}, V_1^{\|S}, E^{|S}, \mathbb{U}, \lambda^{|S})$, where $V_\sigma^{\|S} = \emptyset$, and $V_{1-\sigma}^{\|S} = V$. The main connection between $\mathcal{A}, \mathcal{A}^{|S}$ and $\mathcal{A}^{\|S}$ is that every path in $\mathcal{A}^{|S}$ ($\mathcal{A}^{\|S}$) is a path in $\mathcal{A}$ that is also consistent with S, and vice versa. We can thus reason about paths in $\mathcal{A}^{|S}$ (or $\mathcal{A}^{\|S}$) instead of paths in $\mathcal{A}$ that are consistent with S.

Memoryless determinacy. A game is *memoryless from* v if the player that wins from v has a memoryless strategy that is winning from v.

A game is *point-wise memoryless for Player* σ if the memoryless winning region for Player σ coincides with the winning region for Player σ. A game is *uniform memoryless for Player* σ if there is a memoryless strategy for Player σ that is winning from every vertex in that player's winning region.

A game is *point-wise memoryless determined* if it is determined and it is point-wise memoryless for both players. A game is *uniform memoryless determined* if it is determined and uniform memoryless for both players.

Winning conditions. A *winning condition* is a set $W \subseteq \mathbb{U}^\omega$. If W is a winning condition and $\mathcal{A}$ is an arena, the objective $O^{\mathcal{A}}(W)$ induced by W is defined as follows: $O^{\mathcal{A}}(W) = \{\pi \in plays(\mathcal{A}) : \lambda(\pi) \in W\}$. For ease of readability we may drop the superscript $\mathcal{A}$ and write $O(W)$ instead of $O^{\mathcal{A}}(W)$.

Here are some standard winning conditions:

- The *parity condition* consists of those infinite sequences $c_1c_2\cdots \in \mathbb{N}^\omega$ such that the largest label occurring infinitely often is even.
- For $\nu \in \mathbb{R}$, the *ν-mean-payoff condition* consists of those infinite sequences $c_1c_2\cdots \in \mathbb{R}$ such that $\limsup_{k\to\infty} \frac{1}{k}\sum_{i=1}^{k} c_i$ is at most ν.

[2] For ease of presentation, we consider plays of both finite- and infinite-duration games to be infinite. Obviously, in games finite-duration, games such as FCGs, the winner is determined by a finite prefix of the play, and the moves after this prefix are immaterial.

- The *energy condition*, for a given *initial credit* $r \in \mathbb{N}$, consists of those infinite sequences $c_1 c_2 \cdots \in \mathbb{Z}^\omega$ such that $r + \sum_{i=1}^{k} c_i \geq 0$ for all $k \geq 1$.
- The *energy-parity condition (for a given initial credit r)* is defined as consisting of $(c_1, d_1)(c_2, d_2) \cdots \in \mathbb{N} \times \mathbb{Z}$ such that $c_1 c_2 \cdots$ satisfies the parity condition and $d_1 d_2 \cdots$ satisfies the energy condition with initial credit r.

3. Cycles and games

In this section we define the cycles-decomposition of a path, as well as the first-cycle game, and the infinite-duration all-cycles game.

Cycles-decomposition. A *cycle* in an arena $\mathcal{A}$ is a finite path π such that $\mathsf{start}(\pi) = \mathsf{end}(\pi)$. Note that, like paths, cycles are ordered. Hence, the cycles $(v_1, v_2)(v_2, v_1)$ and $(v_2, v_1)(v_1, v_2)$ are not identical.

Algorithm 1 CycDec(s, π).

Require: s is a finite (possibly empty) simple path ▷ initial stack content
Require: π is a finite or infinite path $\pi_1 \pi_2 \cdots$ ▷ the path to decompose
Require: If s is non-empty then $\mathsf{end}(s) = \mathsf{start}(\pi)$ ▷ $s\pi$ must form a path
  $step = 1$
  while $step \leq |\pi|$ **do** ▷ Start a step
    Append π_{step} to s ▷ Push current edge into stack
    Say $s = e_1 e_2 \cdots e_m$
    if $\exists i : e_i e_{i+1} \cdots e_m$ is a cycle **then** ▷ If stack has a cycle
      Output $e_i e_{i+1} \cdots e_m$ ▷ output the cycle
      $s := e_1 \cdots e_{i-1}$ ▷ Pop the cycle from the stack
    end if
    $step := step + 1$ ▷ advance to next input edge
  end while

The code appearing in Algorithm 1 defines an algorithm CycDec that takes as input a (usually empty) simple path s, which is treated as the initial contents of a stack, and a path π (finite or infinite). At step $j \geq 1$, the edge π_j is pushed onto the stack and if, for some k, the top k edges on the stack form a cycle, this cycle is output, then popped, and the procedure continues to step $j + 1$.

Note that CycDec may take a finite path π and non-empty stack s as input. Moreover, it halts if and only if π is a finite path.

The sequence of cycles output by this procedure when input the empty stack $s = \epsilon$ and path π, denoted $cycles(\pi)$, is called the *cycles-decomposition of* π. The *first cycle of* π, written $first(\pi)$, is the first cycle in $cycles(\pi)$. For example, if

$$\pi = (v, w)(w, x)(x, w)(w, v)(v, s)(s, x)[(x, y)(y, z)(z, x)]^\omega,$$

then

$$cycles(\pi) = (w, x)(x, w), (v, w)(w, v), (x, y)(y, z)(z, x), (x, y)(y, z)(z, x), \ldots,$$

and the first cycle of π is $(w, x)(x, w)$.

It is easy to see that, if the algorithm CycDec is run on a path π in an arena with n nodes, then at most $n - 1$ edges of π are pushed but never popped (like the edges (v, s) and (s, x) in the example above).[3] So we have:

Lemma 1. *For every path π in arena $\mathcal{A}$ with vertex set V, there are at most $|V| - 1$ indices i such that π_i does not appear in any of the cycles in $cycles(\pi)$.*

Given a play π in $\mathcal{A}$, an initial stack content s, and $i \geq 0$, we write $stack^i(s, \pi)$ for the contents of the stack of algorithm CycDec(s, π) at the beginning of step $i + 1$ (i.e., just before the edge π_{i+1} is pushed). Note that if $stack^i(s, \pi)$ is not empty then it ends with the vertex $\mathsf{start}(\pi_{i+1})$, whether or not a cycle was popped during step i. For $i \geq 1$, we define $cycle^i(s, \pi)$ to be the cycle output by the algorithm CycDec(s, π) during step i, or ϵ if no cycle was output at step i. We may drop s when $s = \epsilon$, and we may drop i when $i = |\pi|$. For instance, $stack(\pi)$ is the stack content at the end of the algorithm CycDec(ϵ, π) for a finite π; and $cycle^i(\pi)$ is the cycle output during step i of the algorithm CycDec(ϵ, π).

Given that, for every $i \geq 1$, the behaviour of the algorithm CycDec(s, π) from the end of step i onwards is completely determined by $stack^i(s, \pi)$, and the suffix $\pi_{i+1} \pi_{i+2} \ldots$ of π, the following lemma is immediate:

Lemma 2. *Let π be a play in $\mathcal{A}$, let $i \geq 0$, let $\mathrm{w} = stack^i(\pi)$, and let $\pi' = \pi_{i+1} \pi_{i+2} \ldots$ and $\pi'' = \mathrm{w} \cdot \pi'$. Then for every $j \geq 0$ we have that*

[3] As we show in Section 8, this allows one to reason, for instance, about the initial credit problem for energy games.

1. $stack^{i+j}(\pi) = stack^{j}(\mathrm{w}, \pi') = stack^{|\mathrm{w}|+j}(\pi'')$, *and*
2. $cycle^{i+j}(\pi) = cycle^{j}(\mathrm{w}, \pi') = cycle^{|\mathrm{w}|+j}(\pi'')$.

Furthermore, for every $0 \leq l \leq |\mathrm{w}|$, *we have that* $stack^{l}(\pi'') = \mathrm{w}_1 \dots \mathrm{w}_l$, *and* $cycle^{l}(\pi'') = \epsilon$.

Cycle properties. For a given $\mathbb{U}$, a *cycle property* is a set $Y \subseteq \mathbb{U}^*$, used later on to define winning conditions for games.[4] Here are some cycle properties that we refer to in the rest of the article:

1. Let cyc-EvenLen be those sequences $c_1 c_2 \cdots c_k \in \mathbb{U}^*$ such that k is even.
2. Let cyc-Parity be those sequences $c_1 \cdots c_k \in \mathbb{N}^*$ such that $\max_{1 \leq i \leq k} c_i$ is even.
3. Let cyc-Energy be those sequences $c_1 \cdots c_k \in \mathbb{Z}^*$ such that $\sum_{i=1}^{k} c_i \geq 0$.
4. Let cyc-GoodForEnergy be those sequences $(c_1, d_1) \cdots (c_k, d_k) \in (\mathbb{N} \times \mathbb{Z})^*$ such that $\sum_{i=1}^{k} d_i > 0$, or both $\sum_{i=1}^{k} d_i = 0$ and $c_1 \cdots c_k \in$ cyc-Parity.
5. Let cyc-MeanPayoff$_\nu$ (for $\nu \in \mathbb{R}$) be those sequences $c_1 \cdots c_k \in \mathbb{R}^*$ such that $\frac{1}{k} \sum_{i=1}^{k} c_i \leq \nu$.
6. Let cyc-MaxFirst be those sequences $c_1 \cdots c_k \in \mathbb{N}^*$ such that $c_1 \geq c_i$ for all i with $1 \leq i \leq k$.
7. Let cyc-EndsZero be those sequences $c_1 \cdots c_k \in \mathbb{N}_0^*$ such that $c_k = 0$.

If $Y \subseteq \mathbb{U}^*$ is a cycle property, write $\neg Y$ for the cycle property $\mathbb{U}^* \setminus Y$. We will usually use Y to define goals for Player 0 (hence Player 1's goals would be naturally associated with $\neg Y$). It is convenient to define $Y^0 = Y$, and $Y^1 = \neg Y$. This allows us to refer to the goals of Player σ in terms of Y^σ.

We isolate two important classes of cycle properties (the first is inspired by [4]):

1. Say that Y is *closed under cyclic permutations* if $ab \in Y$ implies $ba \in Y$, for all $a \in \mathbb{U}, b \in \mathbb{U}^*$.
2. Say that Y is *closed under concatenation* if $a \in Y$ and $b \in Y$ imply that $ab \in Y$, for all $a, b \in \mathbb{U}^*$.

Note, for example, that the cycle properties 1–5 above are closed under cyclic permutations and concatenation; and that ¬cyc-EvenLen is closed under cyclic permutations but not under concatenation.

If $\mathcal{A}$ is an arena with labelling λ, and C is a cycle in $\mathcal{A}$, we say that a *cycle* C *satisfies* Y if $\lambda(C) \in Y$.

First-cycle games and all-cycles games.

For arena $\mathcal{A} = (V_0, V_1, E, \mathbb{U}, \lambda)$ and cycle property $Y \subseteq \mathbb{U}^*$, we define two objectives (subsets of $plays(\mathcal{A})$):

1. $\pi \in O^{\mathcal{A}}_{\mathsf{first}}(Y)$ if $first(\pi)$ satisfies Y.
2. $\pi \in O^{\mathcal{A}}_{\mathsf{all}}(Y)$ if every cycle in $cycles(\pi)$ satisfies Y.

To ease readability, we drop the superscript $\mathcal{A}$ when the arena is clear from the context and write, for example, $O_{\mathsf{all}}(Y)$ instead of $O^{\mathcal{A}}_{\mathsf{all}}(Y)$.

The game $(\mathcal{A}, O_{\mathsf{first}}(Y))$ is called a *first-cycle game (over* Y*)*, and the family of all first-cycle games over Y (i.e., taking all possible arenas) is denoted $\textsc{Fcg}(Y)$. Similarly, we write $\textsc{Acg}(Y)$ for the family of *all-cycles games* over Y. For instance, $\textsc{Fcg}$(cyc-Parity) consists of those games such that Player 0 wins iff the largest label occurring on the first cycle is even.

A game $(\mathcal{A}, O)$ is *finitary* if every play is already won after a finite number of steps, i.e., for every $\pi \in plays(\mathcal{A})$ there exists $K_\pi \in \mathbb{N}$ such that, if Player σ wins the play π, then Player σ also wins every play of $\mathcal{A}$ starting with the prefix $\pi[1, K_\pi]$. Clearly FCGs are finitary since the winner is already determined by the first-cycle of the play (recall that arenas are finite). A basic result states that finitary games (of perfect information) are determined.[5] We recap the proof because we use it in Theorem 1 to determine the computational complexity of deciding which player has a winning strategy in a given FCG.

Lemma 3. *Every finitary game* $(\mathcal{A}, O)$ *is determined.*

Proof. Suppose $(\mathcal{A}, O)$ is finitary. Fix a starting node v and note that the set of finite node-paths starting in v form a tree, which we call the *game-tree*. Plays in $\mathcal{A}$ correspond to (infinite) branches in the game-tree. Prune every branch π of the game-tree at its K_πth node to get the pruned game-tree T. By Kőnig's Tree Lemma, the pruned game-tree T is finite since it is finitely branching and contains no infinite paths. Turn T into an And–Or tree: label an internal node ρ by $\vee$ in case $\mathsf{end}(\rho) \in V_0$, and by $\wedge$ in case $\mathsf{end}(\rho) \in V_1$; label a leaf ρ by *true* if every play extending ρ is won by Player 0, and by *false* if every play extending ρ is won by Player 1. Note that every leaf gets a label. It is easy to see that Player 0 has a winning strategy from v in the game $(\mathcal{A}, O)$ if and only if the And–Or tree evaluates to *true*. □

[4] When $\mathbb{U}$ is clear from the context, we will not mention it.

[5] This is usually attributed to Zermelo, but also follows from the fact that open games are determined, e.g., [7].

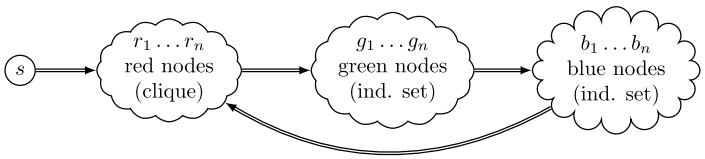

Fig. 1. double arrows represent one edge from every node to every node.

4. First-cycle games

4.1. Memory bounds and complexity

In this section we discuss the amount of memory required to win FCGs and the computational complexity of deciding which player has a winning strategy.

We begin by showing that while strategies with memory $2^{O(|V|\log|V|)}$ always suffice, there are FCGs that require $2^{\Omega(|V|\log|V|)}$ memory.

Proposition 1.

1. *For a FCG on an arena with n vertices, if Player σ wins from v, then every winning strategy for Player σ starting from v uses memory at most $n!$.*
2. *For every $n \in \mathbb{N}$ there exists a FCG on an arena with $3n+1$ vertices, and a vertex v, such that every winning strategy for Player 0 from v uses memory at least $n!$.*

Proof. For the upper bound, note that it is enough to remember the whole history of the play until a cycle is formed, i.e., all histories of length at most $n-1$. To store all histories of length k requires $\sum_{i\leq k} i! < (k+1)!$ many states. Thus, $n!$ memory suffices.

We now turn to the lower bound.

Sketch. We define a game in which Player 1 can select any possible permutation of $\{1,\dots,n\}$ and, in order to win, Player 0 must remember this permutation. Consider the arena in Fig. 1. Intuitively, the red nodes are used by Player 1 to select a permutation of $\{1,\dots,n\}$, and the green and blue nodes are used by her to query Player 0 as to the relative order of any pair of indices i, j in this permutation. The outgoing edges from the blue nodes are used by Player 0 to respond to the query by going back to either r_i or r_j. Player 0 wins the game if the cycle contains both r_i and r_j, hence, he must remember the order of the pair i, j in the permutation. Since this is true for every such pair of indices, he must use at least $n!$ memory to store the permutation. The winning condition captures the above intuition, and also ensures that Player 1 loses if she deviates from the above protocol.

Details. The arena $\mathcal{A} = (V_0, V_1, E, \mathbb{U}, \lambda)$ has n red nodes $r_1,\dots,r_n$, n green nodes $g_1,\dots,g_n$, n blue nodes $b_1,\dots,b_n$, and an initial node s. The red nodes form a clique (i.e., $(r_i, r_j) \in E$ for every $1 \leq i \neq j \leq n$), the green nodes form an independent set, and so do the blue nodes (i.e., $(g_i, g_j) \notin E$ and $(b_i, b_j) \notin E$ for every $1 \leq i, j \leq n$). In addition, for every $1 \leq i, j \leq n$ we have the edges $(s, r_j), (r_i, g_j), (g_i, b_j), (b_i, r_j)$. Player 0 owns the blue nodes ($V_0 = \{b_1,\dots,b_n\}$), and the rest belong to Player 1. In order to correctly describe the winning condition, we label the nodes as follows: for every $1 \leq i \leq n$, the nodes r_i, g_i, b_i are labelled $(Red, i), (Green, i), (Blue, i)$, respectively; and the node s is labelled by 0. In other words, $\mathbb{U} = \{0\} \cup (\{Red, Green, Blue\} \times \{1,\dots,n\})$, and the induced edge labelling is: for every $(v, w) \in E$, we have that $\lambda(v, w) = (Red, i)$ if $v = r_i$, $\lambda(v, w) = (Green, i)$ if $v = g_i$, $\lambda(v, w) = (Blue, i)$ if $v = b_i$, and otherwise $\lambda(v, w) = 0$.

We now define the winning condition. Player 0 wins a play iff the first cycle $(v_1, v_2)\dots(v_k, v_{k+1})$ on the play satisfies one of the following three conditions[6]:

1. the node just before the end is not blue (i.e., $v_k \notin \{b_1,\dots,b_n\}$); or
2. it does not have exactly 1 green node and 1 blue node; or
3. exactly 1 green node g_j, and 1 blue node b_l appear on it, and $b_l = v_k$, and
   (a) it has red nodes labelled with the same numbers as g_j and b_l (i.e., r_j, r_l are on the cycle), and
   (b) it starts with a red node labelled with a number equal to that of the green or the blue node (i.e., $v_1 \in \{r_j, r_l\}$).

Informally, the first two conditions above ensure that Player 0 can win if Player 1 does not select a permutation followed by a query, whereas the third condition ensures that if Player 1 does, then Player 0 can win but only if he remembers the whole permutation.

[6] For simplicity, we state the winning condition in terms of the nodes on the cycle and not in terms of the labels of the edges. However, since we essentially label an edge by the name of its source node, the latter can be easily done.

We first prove that Player 0 has a winning strategy.[7] We distinguish between two types of plays:

(i) Player 1 selects a permutation through the red nodes, and then queries Player 0 by going to some green node g_j, followed by a blue node b_l; i.e., $u = (s, r_{i_1})(r_{i_1}, r_{i_2}) \dots (r_{i_{n-1}}, r_{i_n})(r_{i_n}, g_j)(g_j, b_l)$ are the first $n+2$ edges of the play, and for every $1 \leq x \leq n$ we have that $x = i_h$ for some $1 \leq h \leq n$. In this case, let $1 \leq \tilde{j}, \tilde{l} \leq n$ be such that $i_{\tilde{j}} = j$ and $i_{\tilde{l}} = l$, and observe that Player 0 can win by taking the edge (b_l, r_{i_h}) where $h = min(\tilde{j}, \tilde{l})$ (i.e., by taking the edge back to the node among r_i, r_j that appears sooner on u). Indeed, the first cycle of the play will then be $(r_{i_h}, r_{i_{h+1}}) \dots (r_{i_{n-1}}, r_{i_n})(r_{i_n}, g_j)(g_j, b_l)$, and it will satisfy the third option in the winning condition (in particular, by our choice of h, both r_i and r_j are on the cycle, and $i_h \in \{j, l\}$).

(ii) The play never reaches a blue node, or when the play reaches a blue node for the first time it does not conform to the pattern given in case (i) above. If the play never reaches a blue node then Player 0 wins since the first cycle formed satisfies the first option in the winning condition. This is also true if a cycle was formed before the play reaches a blue node for the first time. Hence, we are left with the option that the prefix of the play is of the form $u = (s, r_{i_1})(r_{i_1}, r_{i_2}) \dots (r_{i_{t-1}}, r_{i_t})(r_{i_t}, g_j)(g_j, b_l)$, where $t < n$, and for $x \neq y$ we have $r_{i_x} \neq r_{i_y}$. In this case, Player 0 first takes the edge (b_l, r_m), where $1 \leq m \leq n$ is the index of a red node that did not yet appear on the play (i.e., $i_x \neq m$ for all $1 \leq x \leq t$). Note that this does not yet form a cycle and the play continues. Now, the game can proceed in three ways, all losing for Player 1: the first is by keeping the play forever away from blue nodes, thus closing a cycle satisfying option 1 in the winning condition; the second is by closing the first cycle by going back to b_l, again losing by option 1 (since then v_k is green); the third is by reaching some blue node b_h different from b_l without yet forming a cycle. In this last case, Player 0 wins (using the second option in the winning condition) by taking the edge (b_h, r_{i_t}) and thus forming a cycle that contains two different blue nodes: b_l, b_h.

We now show that every strategy for Player 0 that uses less than $n!$ memory is not winning. Let ξ be a Player 0 strategy that is implemented using a Mealy machine $\mathcal{M}$ with less than $n!$ states. Hence, there are two different permutations $p = i_1, \dots i_n$ and $p' = i'_1, \dots, i'_n$ of $\{1, \dots, n\}$, such that $\mathcal{M}$ is in the same state m after reading the history $s \cdot r_{i_1} \dots r_{i_n}$, as after reading the history $s \cdot r_{i'_1} \dots r_{i'_n}$. Let h be the smallest index such that $i_h \neq i'_h$, and let $j := i_h$ and $l := i'_h$, and let $1 \leq \tilde{j}, \tilde{l} \leq n$ be such that $i'_{\tilde{j}} = j$ and $i_{\tilde{l}} = l$. Simply put, j appears in position h in p and position $\tilde{j}$ in p', whereas l appears in position h in p' and position $\tilde{l}$ in p. The minimality of h implies that $\tilde{j}, \tilde{l} > h$, and thus l appears after j in p, but before j in p'.

Observe that the two histories $\rho = s \cdot r_{i_1} \dots r_{i_n} \cdot g_j \cdot b_l$, and $\rho' = s \cdot r_{i'_1} \dots r_{i'_n} \cdot g_j \cdot b_l$ also put $\mathcal{M}$ in the same state and thus, ξ responds to both histories with the same move, say by taking the edge (b_l, r_x). Observe that, since p, p' are permutations, every red node already appears on ρ, ρ'. Hence, in both cases, every possible move of Player 0 from b_l closes a cycle that has exactly one green and one blue node, and the blue node appears just before the end. It follows that, in both cases, in order to win Player 0 must satisfy items 3.*a* and 3.*b* of the winning condition. In order to satisfy 3.*a* he must choose $r_x = r_l$ or $r_x = r_j$. However, that prevents him from satisfying item 3.*b* in both cases since r_j appears on ρ before r_l, but after it on ρ'. More formally, consider first the option $r_x = r_l$ and the history ρ. Recall that $i_{\tilde{l}} = l$, that $j = i_h$, and that $\tilde{l} > h$. Hence, the first cycle formed is $(r_{i_{\tilde{l}}}, r_{i_{\tilde{l}+1}}) \dots (r_{i_{n-1}}, r_{i_n})(r_{i_n}, g_j)(g_j, b_l)(b_l, r_{i_{\tilde{l}}})$, and it does not contain the required r_j. Similarly, if $r_x = r_j$ consider the history ρ', and note that the first cycle formed is $(r_{i'_{\tilde{j}}}, r_{i'_{\tilde{j}+1}}) \dots (r_{i'_{n-1}}, r_{i'_n})(r_{i'_n}, g_j)(g_j, b_l)(b_l, r_{i'_{\tilde{j}}})$, and it does not contain the required r_l. It follows that ξ is not a winning strategy. □

We now address the complexity of solving FCGs with efficiently computable cycle properties. Given a game, and a starting node v, solving the game is the problem of deciding whether Player 0 or Player 1 wins from v. We show that this problem is in general Pspace-complete.

Theorem 1.

1. *If Y is a cycle property for which solving membership is in* Pspace *then the problem of solving games in* Fcg(Y) *is in* Pspace.
2. *There exist a cycle property Y, that is computable in linear-time, such that the problem of solving games in* Fcg(Y) *is* Pspace*-hard.*

Proof. Let $\mathcal{A}$ be an arena with n nodes. Following the proof of Lemma 3, for the special case of FCGs, prune a branch of the game-tree once the first cycle is formed, and label the game-tree to get an And–Or tree. Every node in the And–Or tree is of depth at most n, and has at most n children. Hence, evaluating the tree can be done, using a depth-first algorithm, in space polynomial in n plus the maximal space required to compute the labels of the leaves ρ in the tree. Note that the label of a leaf ρ is determined by whether $\lambda(c) \in Y$ or not (where c is the first-cycle on ρ). By our assumption on Y this can be done in polynomial space, hence the upper bound follows.

[7] We define the strategy only for histories that are consistent with it. Obviously, it can be arbitrarily defined on any other history.

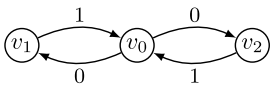

Fig. 2. Sample arena A. Circles are Player 0 nodes, solid lines are edges.

The lower bound follows immediately by reducing the PSPACE-hard problem QBF to solving FCG(Y), where Y is the set of all sequences whose last two elements are identical. The proof uses the same arena (which is vertex-labelled – see Remark 1) as in the reduction of QBF to Generalised Geography (e.g., [11, Theorem 8.11]). To quickly recap, the basic idea there is to view QBF as a game in which Player 0 (resp. Player 1) assigns the values of an existentially (resp. universally) quantified variable x_i by picking a node labelled by the literal x_i or the literal $\neg x_i$; after all variables are thus assigned, Player 1 picks a clause (challenging Player 0 to show that it is true), and finally Player 0 responds by picking a literal of this clause (that he claims is true). The structure of the arena is such that the only outgoing edge from the node picked by Player 0 in this last step is the node labelled by the same literal that was available during the value-assigning phase. Thus, if indeed that literal was picked in the assignment phase, then the next move closes a cycle that satisfies Y and otherwise, taking this edge does not close a cycle, forcing Player 0 to close a losing cycle in the next step. Overall, the QBF formula is true iff Player 0 wins this FCG. □

4.2. The relation between point-wise and uniform memoryless determinacy

In this section we consider the difference between point-wise memoryless determinacy and uniform memoryless determinacy in FCGs. We begin by considering the simple case of solitaire games.

Theorem 2. *All solitaire FCGs are point-wise memoryless determined. However, some solitaire FCG's are not uniform memoryless determined.*

Proof. The fact that a solitaire FCG is point-wise memoryless determined is simply because once a node repeats the game is effectively over, and thus the player makes at most one meaningful move from any node. In other words, Player σ can win from a node v iff there is a play π starting in v such that the first cycle $\pi_i \dots \pi_j$ on π satisfies Y^σ. Traversing the prefix $\pi_1 \dots \pi_j$ requires no memory since each vertex on it is the source of exactly one edge. For the second item, consider the arena in Fig. 2. Observe that, for the cycle property $Y = \mathsf{cyc\text{-}EndsZero}$, no memoryless strategy is winning from both v_1 and v_2. □

We now consider two-player games. In contrast with solitaire games, some two-player FCGs are not point-wise memoryless determined. Indeed, any solitaire game (in which all nodes belong to Player 0) that is not uniform memoryless determined, can be turned into a two player game in which Player 0 wins from some node w, but requires memory to do so: simply add a single Player 1 node w that has outgoing edges to all nodes in the original game.

As we later show (Corollary 1, Page 205), if a cycle property Y is closed under cyclic permutations then all solitaire games in FCG(Y) are uniform memoryless determined. Unfortunately, for two player games, this is not enough even for point-wise memoryless determinacy. Indeed, Theorem 3 shows that closure of Y under cyclic permutations is a necessary condition for having all games in FCG(Y) be point-wise memoryless determined, but it is not a sufficient one.[8]

We also show that, for cycle properties Y that are closed under cyclic permutations, if a game in FCG(Y) is point-wise memoryless for a player then it is also uniform memoryless for this player (Theorem 3 Item 3). The proof of this fact uses the following Lemma.

Lemma 4. *Suppose Y is closed under cyclic permutations, and let S be a strategy for Player σ in arena $\mathcal{A}$. If S is winning from v, then S is winning from every node $w \in reach_{\mathcal{A}}(S, v)$.*

Proof. We begin with the intuition.

Sketch. If c is the first cycle of some play $\pi \in plays(S, w)$, then we can construct a path π' starting in v and consistent with S whose first cycle is some cyclic permutation c' of c, as follows: we proceed along some simple path from v to w until we hit a node on π; if this node is not on c, we continue along π until we reach a node on c; and finally, we cycle along the nodes of c until we return to where we started, forming a cycle c'. Note that c' is not necessarily identical to c since in the first stage we may have hit π somewhere in the middle of c. Since $c' = first(\pi') \in Y^\sigma$, and Y^σ is closed under cyclic permutations (note that Y is closed under cyclic permutations if and only if $\neg Y$ is closed under cyclic permutations), we get that π is won by Player σ.

Details. It is enough to show that in the arena $\mathcal{A}^{|S}$, for every play π starting in w, there is a path π' starting in v, such that the first cycle of π' is a cyclic permutation of the first cycle $\pi_m \dots \pi_n$ of π. We construct π' as follows. Let ρ be a path

[8] This result corrects a mistake in [4] that claims that this is a sufficient condition (note that they do not address the question of whether it is necessary).

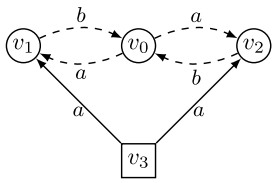

Fig. 3. Sample arena B. Circles are Player 0 nodes, squares are Player 1 nodes, solid lines are edges, dashed lines represent (sequences of edges that are) simple paths.

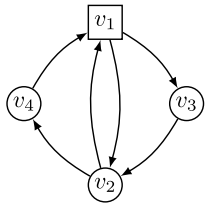

Fig. 4. Sample arena C. Circles are Player 0 nodes, squares are Player 1 nodes, solid lines are edges.

in $\mathcal{A}^{|S}$ from v to w (such a path exists since $w \in \mathit{reach}(S, v)$). Let ℓ be the smallest index such that ρ_ℓ intersects $\pi[1, n]$, i.e., such that $\mathsf{end}(\rho_\ell) = \mathsf{start}(\pi_j)$ for some $j \leq n$. Note that ℓ is well defined since $\mathsf{end}(\rho) = w = \mathsf{start}(\pi_1)$. We consider two cases (depending on whether or not ρ first intersects π before π starts traversing $\mathit{first}(\pi)$).

Case $j \leq m$. Let $\pi' = \rho_1 \dots \rho_\ell \pi_j \dots \pi_n$, and note that $\mathit{first}(\pi') = \mathit{first}(\pi)$.

Case $m < j \leq n$. Let $\pi' = \rho_1 \dots \rho_\ell \pi_j \dots \pi_n \pi_m \dots \pi_{j-1}$, and note that $\mathit{first}(\pi') = \pi_j \dots \pi_n \pi_m \dots \pi_{j-1}$ which is a cyclic permutation of $\mathit{first}(\pi)$. □

Theorem 3.

1. *If Y is not closed under cyclic permutations then there is a game in* $\textsc{Fcg}(Y)$ *that is not point-wise memoryless determined.*
2. *There exists a cycle property Y, closed under cyclic permutations, and a game in* $\textsc{Fcg}(Y)$ *that is not point-wise memoryless determined.*
3. *If a cycle property Y is closed under cyclic permutations, then every game in* $\textsc{Fcg}(Y)$ *that is point-wise memoryless for Player σ is also uniform memoryless for Player σ.*

Proof. For the first item, assume that Y is not closed under cyclic permutations, and let $a, b \in \mathbb{U}^*$ be such that $ab \in Y$ but $ba \notin Y$. Consider the arena in Fig. 3. Observe that Player 0 wins from v_3, but in order to do so he needs to remember if the play arrived to v_0 from v_1 or from v_2.

For the second item, consider the arena in Fig. 4, and the cycle property $Y = \mathsf{cyc\text{-}EvenLen}$. Obviously, Y is closed under cyclic permutations. However, starting at v_1, Player 0 has a winning strategy, but no memoryless winning strategy – when choosing the outgoing edge from v_2 the player needs to remember if the previous node was v_1 (in which case he should return to v_1), or v_3 (in which case he should go to v_4).

For the third item, write $W_\sigma := \mathsf{WR}^\sigma(\mathcal{A}, O)$, and assume that the game is point-wise memoryless for Player σ, and for every $v \in W_\sigma$ let S_v be a memoryless winning strategy for Player σ from v.

Sketch. The idea is to define a memoryless strategy S for Player σ that is winning from every node $v \in W_\sigma$ by considering the nodes of W_σ (in some arbitrary order), and if v is the next node in W_σ for which $S(v)$ is not yet defined, then define $S(w) := S_v(w)$ for all $w \in \mathit{Reach}(S_v, v)$ that have not yet been defined. Thus, S is memoryless by construction. The main work is to show that it is winning (this is where Lemma 4 is used).

Details. Fix some arbitrary linear ordering $v_1 \prec v_2 \prec \dots \prec v_n$ of the nodes in W_σ. We iteratively build a memoryless strategy S for Player σ that is winning from every node $v \in W_\sigma$. At each round $j \geq 0$ of this construction, we write $V^j \subseteq V$ for the set of nodes considered by the end of that round, and have that S is defined for every node in $V^j \cap V_\sigma$. At round 0, we begin with $V^0 = \emptyset$, and with $S(v)$ undefined for all $v \in V_\sigma$. At round $j > 0$, if $V^{j-1} = V$ then we are done, and otherwise we proceed as follows:

(1) If $W_\sigma \not\subseteq V^{j-1}$, let v be the smallest vertex (by the ordering $\prec$) such that $v \in W_\sigma \setminus V^{j-1}$. Set $V^j := V^{j-1} \cup \mathit{reach}(S_v, v)$, and for every $w \in (V_\sigma \cap \mathit{reach}(S_v, v)) \setminus V^{j-1}$ define $S(w) := S_v(w)$.
(2) If $W_\sigma \subseteq V^{j-1}$, then set $V^j = V$, and for every node $v \in V_\sigma \setminus V^{j-1}$ define $S(v)$ arbitrarily. Intuitively, moves from these nodes are unimportant since they are not reachable from any node in W_σ on any play consistent with S.

It is easy to see that S is well defined and memoryless.

It remains to show that S is winning from every $w \in W_\sigma$. The proof is by induction on the round number j. The induction hypothesis is that *(i)* if $w \in V^j$ then $reach(S, w) \subseteq V^j$ and; *(ii)* if $W_\sigma \not\subseteq V^{j-1}$ then S is winning from every $w \in V^j$.

Observe that, by taking the maximal j for which $W_\sigma \not\subseteq V^{j-1}$, item (ii) in the induction hypothesis implies that S is winning from every $w \in W_\sigma$.

For $j = 0$, the hypothesis is trivially true. Assume that the hypothesis holds for all $0 \leq l < j$, and consider round j. If $W_\sigma \subseteq V^{j-1}$, then (i) is true since the construction uses case (2) and sets $V^j = V$, and (ii) is trivially true. If, on the other hand, $W_\sigma \not\subseteq V^{j-1}$ (i.e., the construction uses cases (1)), then let $w \in reach(S_v, v)$ be some node added at round j. Note that to prove that (i) holds, it is enough to show that every node w', that is reachable in $\mathcal{A}^{|S}$ from w, was added before or at round j. In other words, we have to show that $reach(S, w) \subseteq (V^{j-1} \cup reach(S_v, v))$. This can be easily done by inducting on the length of the node-path $\rho = v_1 \dots v_k \in V^*$ from w to w' in $\mathcal{A}^{|S}$. For $k = 0$ this is trivially true. Assume it is true for ρ of length k, and consider ρ of length $k+1$. Now, if ρ_k was added at a previous round, then so did ρ_{k+1} by (i) in the induction hypothesis applied to ρ_k. Otherwise, ρ_k was added at this round (and thus $\rho_k \in reach(S_v, v)$), in which case if ρ_k belongs to the opponent (i.e., $\rho_k \in V_{1-\sigma}$) then all its successors, and in particular ρ_{k+1}, are in $reach(S_v, v)$; and if $\rho_k \in V_\sigma$ then $\rho_{k+1} = S(\rho_k) = S_v(\rho_k) \in reach(S_v, v)$. To complete the proof, we have to show that (ii) holds, i.e., that S is winning from w.

To see that S is winning from w, take any play $\pi \in plays(S, w)$, and let $\pi_m \dots \pi_n$ be the first cycle of π. There are two options: either the prefix $\pi_1 \dots \pi_n$ is consistent with S_v, or it isn't. If it is, since S_v is winning for Player σ from every node in $reach(S_v, v)$ (Lemma 4), and thus in particular from w, we have that π is won by Player σ. Assume then that $\pi_1 \dots \pi_n$ is not consistent with S_v. Note that by our choice of π it is consistent with S and thus, for some $1 \leq h < n$, we have that $S_v(\mathsf{start}(\pi_h)) \neq \mathsf{end}(\pi_h) = S(\mathsf{start}(\pi_h))$. Let $k \leq n$ be the smallest index such that $\mathsf{start}(\pi_k) \in V^{j-1}$. Such a k exists by the above inequality and the fact that (by construction) S agrees with S_v on all nodes added at round j. Observe that if $k > m$ then all the nodes appearing on $\pi_m \dots \pi_n$ are in $reach(S, \pi_k)$, simply by following the cycle $\pi_k \dots \pi_n \pi_m \dots \pi_{k-1}$. Hence, since by item (i) in the induction hypothesis $reach(S, \mathsf{start}(\pi_k)) \subseteq V^{j-1}$, the minimality of k implies that $k \leq m$. We conclude that the suffix $\pi' = \pi_k \pi_{k+1} \dots$ of π, which is a play consistent with S starting from $\mathsf{start}(\pi_k) \in V^{j-1}$, satisfies $first(\pi') = first(\pi)$. By item (ii) in the induction hypothesis, π' is won by Player σ, hence so is π, which completes the proof. □

Theorems 2 and 3 give us the following corollary:

Corollary 1. *If a cycle property Y is closed under cyclic permutations, then every solitaire game in* $\textsc{Fcg}(Y)$ *is uniform memoryless determined.*

5. Memoryless determinacy of first-cycle games

In this section we give a necessary and sufficient condition for a FCG to be memoryless determined. In Section 7 we give an easy-to check condition that is sufficient, but not necessary.

5.1. A full characterisation of memoryless determinacy of all games in $\textsc{Fcg}(Y)$

We begin by introducing some useful shorthand notation. Given an arena $\mathcal{A} = (V_0, V_1, E, \mathbb{U}, \lambda)$, and a node $z \in V$, for a path $\pi \in E^* \cup E^\omega$, define $N_z(\pi) \in \mathbb{N} \cup \{\infty\}$ to be the index of the first edge that starts with z, if one exists. Formally, $N_z(\pi) := \infty$ if z does not occur on π, and otherwise $N_z(\pi) := \min\{j : \mathsf{start}(\pi_j) = z\}$. Also, define $head_z(\pi) := \pi[1, N_z(\pi) - 1]$ to be the prefix of π before $N_z(\pi)$, and $tail_z(\pi) := \pi[N_z(\pi), |\pi|]$ to be the suffix of π starting at $N_z(\pi)$. By convention, if $N_z(\pi) = \infty$ then $head_z(\pi) = \pi$ and $tail_z(\pi) = \epsilon$.

We now define a game, that is very similar to the first-cycle game, except that one of the nodes of the arena is designated as a "reset" node:

Definition 1. Fix an arena $\mathcal{A}$, a vertex $z \in V$, and a cycle property Y. Define the objective $O^{\mathcal{A}}_{z\text{-first}}(Y) \subseteq plays(\mathcal{A})$ to consist of all plays π satisfying the following property: if $head_z(\pi)$ is not a simple path then $first(\pi) \in Y$, and otherwise $first(tail_z(\pi)) \in Y$.

Playing the game with objective $O_{z\text{-first}}(Y)$ is like playing the first-cycle game over Y, however, if no cycle is formed before reaching z for the first time, the prefix of the play up to that point is ignored. Thus, in a sense, the game is reset. Also note that if play starts from z, then the game reduces to a first-cycle game. It turns out that we may assume that a strategy of $(\mathcal{A}, O_{z\text{-first}}(Y))$ makes the same move every time it reaches z:

Definition 2 *(Forgetful at z from v).* For an arena $\mathcal{A}$, a vertex $v \in V$, a Player $\sigma \in \{0, 1\}$, and a vertex $z \in V_\sigma$ belonging to Player σ, we call a strategy T for Player σ *forgetful at z from v* if there exists $z' \in V$ such that $(z, z') \in E$ and for all $\pi \in plays(T, v)$, and all $n \in \mathbb{N}$, if $\mathsf{start}(\pi_n) = z$ then $\mathsf{end}(\pi_n) = z'$.

Lemma 5 *(Forgetful at z from v). Fix an arena $\mathcal{A}$, a vertex $v \in V$, a Player $\sigma \in \{0,1\}$, and a vertex $z \in V_\sigma$ belonging to Player σ. In the game $(\mathcal{A}, O_{z\text{-first}}(Y))$, if Player σ has a strategy S that is winning from v, then Player σ has a strategy T that is winning from v and that is forgetful at z from v.*

Proof. We begin with the intuition.

Sketch. The second time z appears on a play, the winner is already determined, and so the strategy is free to repeat the first move it made at z. On the other hand, when a play visits z the first time, the strategy can make the same move regardless of the history of the play before z, because the winning condition ignores this prefix.

Details. We may suppose that z appears on some play of $plays(S, v)$, otherwise we can take T to be S. Let ρ be a simple path from v to z that is consistent with S. Let $z' := S(\rho)$. Define the strategy T as follows:

$$T(u) := \begin{cases} S(u) & \text{if } z \text{ does not appear on } u, \\ z' & \text{if } \mathsf{end}(u) = z, \\ S(\rho \cdot tail_z(u)) & \text{otherwise.} \end{cases}$$

By definition, T is forgetful at z from v. It remains to show that every $\pi \in plays(T, v)$ is won by Player σ.

First consider the case that $head_z(\pi)$ is not a simple path. By definition, S and T agree on $head_z(\pi)$, and since S is winning, the first cycle on $head_z(\pi)$ (and thus also on π) satisfies Y^σ, and π is won by Player σ.

Now consider the case that $head_z(\pi)$ is a simple path. We need to show that $first(tail_z(\pi))$ is in Y^σ. Define $\pi' := \rho \cdot tail_z(\pi)$. It is easy to see that π' is consistent with T. We claim that $\pi' \in plays(S, v)$. Indeed, the prefix $\rho \cdot (z, z')$ is consistent with S; and for every $j \geq |\rho| + 1$, such that $\mathsf{end}(\pi'_j) \in V_\sigma$, we have, by the third case in the definition of T, that $T(\pi'[1, j]) = S(\rho \cdot tail_z(\pi'[1, j])) = S(\pi'[1, j])$ (the second equality holds since, by the definition of $tail_z$, $\rho \cdot tail_z(\pi'[1, j]) = \pi'[1, j]$). Now, since π' is consistent with T, we have that $T(\pi'[1, j]) = \mathsf{end}(\pi'_{j+1})$, and thus $S(\pi'[1, j]) = \mathsf{end}(\pi'_{j+1})$. This completes the proof of the claim.

To finish the proof, note that $head_z(\pi')$ is a simple path (it is ρ), and that $tail_z(\pi') = tail_z(\pi)$. Hence, since $head_z(\pi')$ is a simple path, and S is winning from v, we know that $first(tail_z(\pi'))$ satisfies Y^σ. Thus, since $head_z(\pi)$ is a simple path and $first(tail_z(\pi))$ satisfies Y^σ, we can conclude that π is won by Player σ. □

We now define the basic notion behind our necessary and sufficient condition for games in $\textsc{Fcg}(Y)$ to be memoryless determined.

Definition 3. For an arena $\mathcal{A}$, and a node v in $\mathcal{A}$, say that $\mathcal{A}$ is *Y-resettable from v* if for every node z, the same player wins from v in both $(\mathcal{A}, O_{\mathsf{first}}(Y))$ and $(\mathcal{A}, O_{z\text{-}\mathsf{first}}(Y))$.

First, we show that Y-resettability is a sufficient condition for having memoryless strategies.

Theorem 4. *Given an arena $\mathcal{A}$, if a node v is such that every sub-arena of $\mathcal{A}$ is Y-resettable from v, then the game $(\mathcal{A}', O_{first}(Y))$ is memoryless from v for every sub-arena $\mathcal{A}'$ of $\mathcal{A}$.*

Proof. A node $z \in V$ is a *choice node* of an arena $\mathcal{B} = (V_0, V_1, E^{\mathcal{B}}, \mathbb{U}, \lambda)$, if there are at least two distinct vertices $v', v'' \in V$ such that $(z, v') \in E^{\mathcal{B}}$ and $(z, v'') \in E^{\mathcal{B}}$.

Sketch. Fix a sub-arena $\mathcal{A}'$ of $\mathcal{A}$. Suppose Player σ has a winning strategy from v in $\mathcal{A}'$. We induct on the number of choice nodes of Player σ. Let z be a choice node for Player σ (if there are none, the result is immediate). Since $\mathcal{A}'$ is Y-resettable from v, Player σ also wins the game with objective $O_{z\text{-}\mathsf{first}}(Y)$ from v. By Lemma 5, Player σ has a strategy S that is winning from v and that is also forgetful at z from v. Thus we may form a sub-arena $\mathcal{B}$ of $\mathcal{A}'$ (and hence of $\mathcal{A}$) by removing all edges from z that are not taken by S. Observe that S is winning from v in $(\mathcal{B}, O_{z\text{-}\mathsf{first}}(Y))$. Since the sub-arena $\mathcal{B}$ is Y-resettable from v, Player σ also wins $(\mathcal{B}, O_{\mathsf{first}}(Y))$ from v. But $\mathcal{B}$ has less choice nodes for Player σ, and thus, by induction, Player σ has a memoryless winning strategy from v in $(\mathcal{B}, O_{\mathsf{first}}(Y))$. This memoryless strategy is also winning from v in $\mathcal{A}'$.

Details. Fix a sub-arena $\mathcal{A}'$ of $\mathcal{A}$, and let $\sigma \in \{0, 1\}$ be such that $v \in \mathsf{WR}^\sigma(\mathcal{A}', O_{\mathsf{first}}(Y))$. The nth inductive hypothesis states: for every sub-arena $\mathcal{B}$ (of $\mathcal{A}'$), in which Player σ has exactly n choice nodes, if Player σ has a winning strategy from v in $(\mathcal{B}, O_{\mathsf{first}}(Y))$, then Player σ has a *memoryless* winning strategy from v in $(\mathcal{B}, O_{\mathsf{first}}(Y))$.

The base case is when $n = 0$, i.e, Player σ has no choice nodes in $\mathcal{B}$. In this case, Player σ has a single strategy: given a history $u \in H_\sigma(\mathcal{B})$ with $end(u) = w$, then take the unique edge $(w, w') \in E^{\mathcal{B}}$. Note that this strategy is memoryless.

Let $n > 0$, and suppose the inductive hypothesis holds for $n - 1$. We will prove it holds for n. To this end, let $\mathcal{B} = (V_0, V_1, E^{\mathcal{B}}, \mathbb{U}, \lambda)$ be a sub-arena (of $\mathcal{A}'$) with n choice nodes for Player σ. Fix one such choice node, and call it z. Suppose Player σ has a winning strategy (not necessarily memoryless) from v in $(\mathcal{B}, O_{\mathsf{first}}(Y))$. Since, by assumption, $\mathcal{B}$ is Y-resettable from v, there is also a winning strategy S for Player σ from v in $(\mathcal{B}, O_{z\text{-}\mathsf{first}}(Y))$. We will use S to prove Player σ has a memoryless winning strategy from v in $(\mathcal{B}, O_{\mathsf{first}}(Y))$. By Lemma 5, we may assume that S is forgetful at

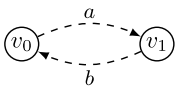

Fig. 5. The dashed lines represent simple paths.

z from v, i.e., there exists $z' \in V$ such that $(z, z') \in E^{\mathcal{B}}$ and for all $\pi \in plays_{\mathcal{B}}(S, v)$, and all $i \in \mathbb{N}$, if $\mathsf{start}(\pi_i) = z$ then $\mathsf{end}(\pi_i) = z'$.

Define the sub-arena $\mathcal{B}_z$ to be the same as $\mathcal{B}$ but with all edges out of z removed except for (z, z'). That is, $\mathcal{B}_z := (V_0, V_1, E_z, \mathbb{U}, \lambda')$ where $E_z := E^{\mathcal{B}} \setminus \{(z, x) : x \neq z'\}$ and λ' is λ restricted to E_z. Then S is well-defined on $\mathcal{B}_z$ (i.e., by our assumption it never says to move from z to a node other than z'). Since S is winning for Player σ from v in the game $(\mathcal{B}, O_{z\text{-first}}(Y))$ and node z belongs to Player σ, conclude that S is winning for Player σ from v in $(\mathcal{B}_z, O_{z\text{-first}}(Y))$. Being a sub-arena of $\mathcal{A}$, by assumption, the arena $\mathcal{B}_z$ is Y-resettable from v. Hence, Player σ wins from v in $(\mathcal{B}_z, O_{\mathsf{first}}(Y))$. Since $\mathcal{B}_z$ is a sub-arena of $\mathcal{A}'$ and has $n-1$ choice nodes for Player σ, we can apply the induction hypothesis to $\mathcal{B}_z$ and obtain that Player σ has a memoryless strategy S_{mem} winning from v in $(\mathcal{B}_z, O_{\mathsf{first}}(Y))$. Recall that $\mathcal{B}_z$ was obtained from $\mathcal{B}$ by removing outgoing edges from $z \in V_\sigma$ (i.e., by providing Player σ with less freedom of movement), and conclude that S_{mem} must be winning for Player σ from v in the game $(\mathcal{B}, O_{\mathsf{first}}(Y))$. This completes the inductive step. □

It is worth noting that the assumption in Theorem 4, that every sub-arena of $\mathcal{A}$ is Y-resettable from v, cannot be replaced by the weaker requirement that only $\mathcal{A}$ is Y-resettable from v. Consider for example the arena $\mathcal{A}$ in Fig. 3 (page 204), and a cycle property $Y \subseteq \mathbb{U}^*$, such that $ab \in Y$ but $ba \notin Y$, for some $a, b \in \mathbb{U}^*$. As argued in Theorem 3, the game $(\mathcal{A}, O_{\mathsf{first}}(Y))$ is not memoryless starting at v_3. While the sub-arena $\mathcal{A}'$, obtained by dropping the edge (v_0, v_1), is not Y-resettable from v_3 (since for $z = v_0$ Player 0 wins $(\mathcal{A}', O_{z\text{-first}}(Y))$ but $(\mathcal{A}', O_{\mathsf{first}}(Y))$ is won by Player 1) the arena $\mathcal{A}$ is Y-resettable from v_3.[9]

Before we provide a converse for Theorem 4, we show that an additional assumption, namely that Y is closed under cyclic permutations, is needed:

Lemma 6. *If Y is not closed under cyclic permutations then there is an arena $\mathcal{A}$, and a node v, for which the game $(\mathcal{A}', O_{first}(Y))$ is memoryless from v for every sub-arena $\mathcal{A}'$ of $\mathcal{A}$, but $\mathcal{A}$ is not Y-resettable from v.*

Proof. Assume that Y is not closed under cyclic permutations, and let $a, b \in \mathbb{U}^*$ be such that $ab \in Y$ but $ba \notin Y$. Consider the arena in Fig. 5, containing only Player 0 nodes, where the path from v_0 to v_1 is labelled by a, and the path from v_1 to v_0 is labelled by b. Observe that $\mathcal{A}$ has only one sub-arena (itself), and there is a single strategy possible in the game $(\mathcal{A}, O_{\mathsf{first}}(Y))$, and it is memoryless. However, $\mathcal{A}$ is not Y-resettable from v_0 since, starting at v_0, Player 0 wins the game $(\mathcal{A}, O_{\mathsf{first}}(Y))$ but not the game $(\mathcal{A}, O_{z\text{-first}}(Y))$ for $z = v_1$. □

We now prove that if Y is closed under cyclic permutations then the converse of Theorem 4 is also true.

Theorem 5. *Let Y be closed under cyclic permutations, let $\mathcal{A}$ be an arena, and let v be a node such that the game $(\mathcal{B}, O_{first}(Y))$ is memoryless from v for every sub-arena $\mathcal{B}$ of $\mathcal{A}$. Then every sub-arena $\mathcal{B}$ of $\mathcal{A}$ is Y-resettable from v.*

Proof. We prove that, for every arena $\mathcal{B}$ and a node v in it, if $(\mathcal{B}, O_{\mathsf{first}}(Y))$ is memoryless from v then $\mathcal{B}$ is Y-resettable from v. The theorem follows by taking $\mathcal{B}$ to be any sub-arena of $\mathcal{A}$.

Let z be a node in $\mathcal{B}$. Lemma 3 (Page 200) implies that FCGs are determined. Thus, it is enough to show that if Player σ wins from v in the game $(\mathcal{B}, O_{\mathsf{first}}(Y))$, then he wins from v in the game $(\mathcal{B}, O_{z\text{-first}}(Y))$. So, fix a Player $\sigma \in \{0, 1\}$ and assume that Player σ wins from v in $(\mathcal{B}, O_{\mathsf{first}}(Y))$. Since by our assumption this game is memoryless determined, there is a memoryless strategy S for Player σ winning from v in $(\mathcal{B}, O_{\mathsf{first}}(Y))$. Consider the sub-arena $\mathcal{B}^{\|S}$ induced by S, and recall that every path in $\mathcal{B}^{\|S}$ is consistent with S. We claim that every simple cycle $\pi = \pi_1 \dots \pi_k$ (of some length k) in $\mathcal{B}^{\|S}$, that is reachable from v, satisfies Y^σ. Let ρ be a path in $\mathcal{B}^{\|S}$ of minimal length that starts in v and ends in $\mathsf{start}(\pi_i)$ for some $1 \leq i \leq k$. Consider the path $\pi' = \rho\pi_i \dots \pi_k\pi_1 \dots \pi_{i-1}$. Since $\pi' \in plays_{\mathcal{B}}(S, v)$, and S is winning from v, we have that the first cycle c on π' satisfies Y^σ. Observe that (by our choice of ρ) $c = \pi_i \dots \pi_k\pi_1 \dots \pi_{i-1}$, and is thus a cyclic permutation of π. Since Y^σ is closed under cyclic permutations (recall that if Y is closed under cyclic permutations then so is $\neg Y$) then π satisfies Y^σ, and the claim is true. In other words, for every play $\rho' \in plays_{\mathcal{B}}(S, v)$, and every simple cycle $c' = \rho'_i \dots \rho'_j$ (for some $i, j \in \mathbb{N}$) on ρ', we have that c' satisfies Y^σ. Hence, S is also winning from v in the game $(\mathcal{B}, O_{z\text{-first}}(Y))$. □

If an arena $\mathcal{A}$ is Y-resettable from v, for every node v, then we simply say that it is Y-resettable. In other words:

[9] One can also come up with such an example (albeit a more complicated one) for $Y = \mathsf{cyc\text{-}EvenLen}$ which is closed under cyclic permutations.

Definition 4. An arena $\mathcal{A}$ is *Y-resettable* if for every $\sigma \in \{0,1\}$, and every node z, we have that $\mathsf{WR}^\sigma(\mathcal{A}, O_{z\text{-first}}(Y)) = \mathsf{WR}^\sigma(\mathcal{A}, O_{\mathsf{first}}(Y))$.

We conclude with this full characterisation:

Theorem 6 *(Memoryless determinacy characterisation of FCGs). The following are equivalent for every cycle property Y:*

1. *Y is closed under cyclic permutations, and every arena $\mathcal{A}$ is Y-resettable.*
2. *Every game in $\textsc{Fcg}(Y)$ is uniform memoryless determined.*

Proof. Suppose Y is closed under cyclic permutations. Theorem 3 (part 3) says that every game in $\textsc{Fcg}(Y)$ that is point-wise memoryless determined is uniform memoryless determined. But since every arena is assumed Y-resettable from every v, Theorem 4 implies that every game in $\textsc{Fcg}(Y)$ is point-wise memoryless determined.

Conversely, suppose every game in $\textsc{Fcg}(Y)$ is uniform memoryless determined. By Theorem 3 (part 1), Y must be closed under cyclic permutations. This also means we can apply Theorem 5, and conclude that every arena is Y-resettable. □

6. Strategy Transfer Theorem: infinite-duration cycle games and first-cycle games

In this section we define the connection between first-cycle games and games of infinite duration (such as parity games, etc.), namely the concept of Y-greedy games. We then prove the Strategy Transfer Theorem, which says, roughly, that for every arena, the winning regions of the First-Cycle Game over Y and a Y-greedy game coincide, and that memoryless winning strategies transfer between these two games.

Definition 5 *(Greedy).* Say that a *game* $(\mathcal{A}, O)$ *is Y-greedy* if

$$O^{\mathcal{A}}_{\mathsf{all}}(Y) \subseteq O \text{ and } O^{\mathcal{A}}_{\mathsf{all}}(\neg Y) \subseteq E^\omega \setminus O.$$

Intuitively, a game $(\mathcal{A}, O)$ is Y-greedy means that Player 0 can win the game $(\mathcal{A}, O)$ if he ensures that every cycle in the cycles-decomposition of the play is in Y, and Player 1 can win if she ensures that every cycle in the cycles-decomposition of the play is not in Y.

An equivalent formulation is that $(\mathcal{A}, O)$ is Y-greedy if

$$O^{\mathcal{A}}_{\mathsf{all}}(Y) \subseteq O \subseteq O^{\mathcal{A}}_{\mathsf{exist}}(Y),$$

where $O^{\mathcal{A}}_{\mathsf{exist}}(Y)$ consists of all plays π such that some cycle in $\mathit{cycles}(\pi)$ satisfies Y.

Here are some examples.

1. Every all-cycles game $(\mathcal{A}, O_{\mathsf{all}}(Y))$ is Y-greedy.
2. Every parity game is cyc-Parity-greedy.
3. Every game with ν-mean-payoff winning condition is $\mathsf{cyc\text{-}MeanPayoff}_\nu$-greedy.

Before proving the Strategy Transfer Theorem we need a lemma that states that one can pump a strategy S that is winning for the first-cycle game to get a strategy $S^{\circlearrowleft}$ that is winning for the all-cycles game by following S until a cycle is formed, removing that cycle from the history, and continuing. The fact that every winning strategy in the first-cycle game of Y can be pumped to obtain a winning strategy in a Y-greedy game, is why we call such games "greedy".

Recall from the Definitions that for a finite path $\pi \in E^*$, the stack content at the end of $\textsc{CycDec}(\epsilon, \pi)$ (Algorithm 1) is denoted $\mathit{stack}(\pi)$. Recall our notational abuse (Page 198) that for a finite path ρ, we may write $S(\rho)$ instead of $S(\mathsf{nodes}(\rho))$.

Definition 6 *(Pumping strategy).* Fix an arena $\mathcal{A}$, a Player $\sigma \in \{0,1\}$, and a strategy S for Player σ. Let the *pumping strategy of* S be the strategy $S^{\circlearrowleft}$ for Player σ, defined, on any input $u = v_1 \dots v_k \in H_\sigma(\mathcal{A})$, as follows:

a. $S^{\circlearrowleft}(u) := S(v_k)$ if $k = 1$ or $\mathit{stack}(\pi) = \epsilon$, and otherwise
b. $S^{\circlearrowleft}(u) = S(\mathit{stack}(\pi))$,

where $\pi \in E^*$ is the path corresponding to u, i.e., $\mathsf{nodes}(\pi) = u$.

Note that $S^{\circlearrowleft}$ is well-defined since if $\mathit{stack}(\pi) \neq \epsilon$ then $\mathit{stack}(\pi)$ ends with $v_k \in V_\sigma$ and so $\mathit{stack}(\pi)$ is in the domain of S. Also note that if S is memoryless then the pumping strategy $S^{\circlearrowleft} = S$.

Lemma 7. *Let $\mathcal{A}, \sigma, S$ and $S^{\circlearrowleft}$ be as in Definition 6, and let $v \in V$. If $\pi \in \mathit{plays}(S^{\circlearrowleft}, v)$ then for every cycle C in $\mathit{cycles}(\pi)$ there exists a finite path ρ consistent with S and starting with v such that:*

- *The first cycle on ρ is C (thus, in particular, if S is winning from v in the game $(\mathcal{A}, O_{first}(Y))$ then C satisfies Y^σ);*
- *ρ only contains edges from π.*

Proof. Sketch. The strategy $S^{\circlearrowleft}$ says to follow S, and when a cycle is popped by CycDec, remove that cycle from the history and continue. Thus, for every cycle C that is popped, let l be the time at which the first edge of C is being pushed, and note that the stack up to time l followed by C is a path consistent with S whose first cycle is C.

Details. Fix $\pi \in plays(S^{\circlearrowleft}, v)$. Every cycle $C \in cycles(\pi)$ was output by the run of CycDec(ϵ, π) at some time $j \in \mathbb{N}$, and is thus of the form $C = cycle^j(\pi)$. Let $\rho := stack_{j-1}(\pi) \cdot \pi_j$, and observe that C is the first cycle on ρ, and the second item in the statement of the lemma is true.

For the first item, it is sufficient to prove, for all $k \in \mathbb{N}$, that $stack_{k-1}(\pi) \cdot \pi_k$ is consistent with S and starts with v. We remind the reader that, by definition, ρ is consistent with S iff:

1. $S(\mathsf{start}(\rho_1)) = \mathsf{end}(\rho_1)$ if $\mathsf{start}(\rho_1) \in V_\sigma$, and
2. $S(\rho[1,n]) = \mathsf{end}(\rho_{n+1})$ for $1 \leq n < |\rho|$ with $\mathsf{end}(\rho[1,n]) \in V_\sigma$.

We prove that $stack_{k-1}(\pi) \cdot \pi_k$ is consistent with S and starts with v, by induction on $k \in \mathbb{N}$.

The base is when $k = 1$. Since $stack_0(\pi) = \epsilon$, we show that the path consisting of the single edge π_1, which starts with v by definition, is consistent with S. Note that we are in item 1. of the consistency condition with $\rho = \pi_1$. So suppose $\mathsf{start}(\pi_1) \in V_\sigma$. Then:

$$\begin{aligned} \mathsf{end}(\pi_1) &= S^{\circlearrowleft}(\mathsf{start}(\pi_1)) && \text{since } \pi \text{ is consistent with } S^{\circlearrowleft}, \\ &= S(\mathsf{start}(\pi_1)) && \text{by definition of } S^{\circlearrowleft}, \text{ part a).} \end{aligned}$$

For the inductive step, let $k > 1$ and suppose the inductive hypothesis holds for $k-1$. We prove it holds for k. There are two cases.

i) Suppose $stack_{k-1}(\pi) = \epsilon$. To show π_k is consistent with S, note that we are again in item 1. of the consistency condition, but this time with $\rho = \pi_k$. Repeat the argument in the base case with π_k replacing π_1. To show that π_k starts with v argue as follows. The fact that $stack_{k-1}(\pi) = \epsilon$ means that after pushing π_{k-1} onto the stack during step $k-1$ of the algorithm CycDec(ϵ, π), the resulting stack forms a cycle. In other words, $\mathsf{end}(\pi_{k-1}) = \mathsf{start}(stack_{k-2}(\pi) \cdot \pi_{k-1})$. But $\mathsf{start}(\pi_k) = \mathsf{end}(\pi_{k-1})$ since π is a path, and $\mathsf{start}(stack_{k-2}(\pi) \cdot \pi_{k-1}) = v$ by the induction hypothesis. Thus $\mathsf{start}(\pi_k) = v$.

ii) Suppose $stack_{k-1}(\pi) \neq \epsilon$. The induction hypothesis states that the path $stack_{k-2}(\pi) \cdot \pi_{k-1}$ is consistent with S and starts with v. Observe that $stack_{k-1}(\pi)$ is a (non-empty) prefix of $stack_{k-2}(\pi) \cdot \pi_{k-1}$. So, $stack_{k-1}(\pi)$ starts with v, and, being a prefix of a path consistent with S, is consistent with S. Thus we only need to consider π_k. That is, we are in item 2. of the consistency condition, with $\rho = stack_{k-1}(\pi) \cdot \pi_k$ and $n = |stack_{k-1}(\pi)|$. If $\mathsf{end}(\rho[1,n]) \in V_\sigma$ then

$$\begin{aligned} \mathsf{end}(\rho_{n+1}) &= \mathsf{end}(\pi_k) && \text{since } \rho_{n+1} = \pi_k, \\ &= S^{\circlearrowleft}(\pi[1,k-1]) && \text{since } \pi \text{ is consistent with } S^{\circlearrowleft}, \\ &= S(stack_{k-1}(\pi)) && \text{by definition of } S^{\circlearrowleft}, \text{ part b),} \\ &= S(\rho[1,n]) && \text{since } \rho[1,n] = stack_{k-1}(\pi). \end{aligned}$$

This completes the inductive step and the proof. □

Corollary 2. *Fix Player $\sigma \in \{0,1\}$ and let $(\mathcal{A}, O)$ be a Y-greedy game. If S is a strategy for Player σ that is winning from v in $(\mathcal{A}, O_{first}(Y))$ then $S^{\circlearrowleft}$ is winning from v in $(\mathcal{A}, O)$.*

Proof. Suppose S is winning from v in the game $(\mathcal{A}, O_{\mathsf{first}}(Y))$. Then, by Lemma 7, for every play $\pi \in plays(S^{\circlearrowleft}, v)$, every cycle in $cycles(\pi)$ satisfies Y^σ, i.e., $\pi \in O^{\mathcal{A}}_{\mathsf{all}}(Y^\sigma)$. By definition of Y-greedy, this means that $S^{\circlearrowleft}$ is winning from v in the game $(\mathcal{A}, O)$. □

We now have the ingredients for the proof of the Strategy Transfer Theorem:

Theorem 7 *(Strategy transfer). Let $(\mathcal{A}, O)$ be a Y-greedy game, and let $\sigma \in \{0,1\}$.*

1. *The winning regions for Player σ in the games $(\mathcal{A}, O)$ and $(\mathcal{A}, O_{first}(Y))$ coincide.*
2. *For every memoryless strategy S for Player σ, and vertex $v \in V$ in arena $\mathcal{A}$: S is winning from v in the game $(\mathcal{A}, O)$ if and only if S is winning from v in the game $(\mathcal{A}, O_{first}(Y))$.*

Proof. Let $Y \subseteq \mathbb{U}^*$ be a cycle property and $\mathcal{A}$ an arena. Suppose that $(\mathcal{A}, O)$ is Y-greedy. We begin by proving the first item. Use Corollary 2 to get that for $\sigma \in \{0, 1\}$,

$$\mathsf{WR}^\sigma(\mathcal{A}, O_{\mathsf{first}}(Y)) \subseteq \mathsf{WR}^\sigma(\mathcal{A}, O).$$

Since first-cycle games are determined (Lemma 3), the winning regions $\mathsf{WR}^0(\mathcal{A}, O_{\mathsf{first}}(Y))$ and $\mathsf{WR}^1(\mathcal{A}, O_{\mathsf{first}}(Y))$ partition V. Thus, since $\mathsf{WR}^0(\mathcal{A}, O)$ and $\mathsf{WR}^1(\mathcal{A}, O)$ are disjoint, the containments above are equalities, as required for item 1.

We prove the second item. Since $S = S^{\circlearrowleft}$ if S is memoryless, conclude by Corollary 2: if S is winning from v in the game $(\mathcal{A}, O_{\mathsf{first}}(Y))$ then it is winning from v in the game $(\mathcal{A}, O)$. For the other direction, assume by contraposition that S is not winning from v in the game $(\mathcal{A}, O_{\mathsf{first}}(Y))$. Since S is memoryless, plays of $\mathcal{A}$ consistent with S are exactly infinite paths in the induced sub-arena $\mathcal{A}^{\|S}$. Hence, there is a path π in the induced solitaire arena $\mathcal{A}^{\|S}$ for which the first cycle, say $\pi[i, j]$, satisfies $Y^{1-\sigma}$. Define the infinite path $\pi' := \pi[1, i-1] \cdot (\pi[i, j])^\omega$ and note that, being a path in $A^{\|S}$, it is a play of $\mathcal{A}$ consistent with S. Moreover, π' has the property that every cycle in its cycles-decomposition (i.e., $\pi[i, j]$) satisfies $Y^{1-\sigma}$. Since $(\mathcal{A}, O)$ is Y-greedy, S is not winning from v in the game $(\mathcal{A}, O)$. □

Since FCGs are determined (Lemma 3, Page 200) use Theorem 7 to get:

Corollary 3. *Every Y-greedy game $(\mathcal{A}, O)$ is determined, has the same winning regions as $(\mathcal{A}, O_{first}(Y))$, and is point-wise (uniform) memoryless determined if and only if the game $(\mathcal{A}, O_{first}(Y))$ is point-wise (uniform) memoryless determined.*

We now state our main result concerning Y-greedy games.

Theorem 8 *(Memoryless determinacy characterisation of greedy games). For every cycle property Y, the following are equivalent:*

1. *Y is closed under cyclic permutations, and every arena is Y-resettable.*
2. *Every Y-greedy game is uniform memoryless determined.*

Proof. This follows from Corollary 3, Theorem 6, and the fact that for every arena $\mathcal{A}$ there is a Y-greedy game with arena $\mathcal{A}$, for example, the game $(\mathcal{A}, O_{\mathsf{all}}(Y))$. □

7. An easy to check sufficient condition on Y ensuring memoryless determinacy of FCGs

As we have seen Y-resettability together with closure under cyclic permutations provides a full characterization of those cycle properties Y for which the games in $\mathrm{Fcg}(Y)$, as well as any Y-greedy games, are memoryless determined. Even though, in many cases, checking whether Y is such that every arena $\mathcal{A}$ (or just the arena(s) of interest) is Y-resettable is not too difficult, we believe that in practice it is much easier to use the following condition on Y which avoids reasoning about arenas altogether. The condition we suggest is the following:

$$Y \text{ and } \neg Y \text{ are closed under concatenation.}$$

As we later show, this condition (together with closure under cyclic permutations) ensures that every arena $\mathcal{A}$ is Y-resettable, and thus by Theorem 6, that all games in $\mathrm{Fcg}(Y)$ are memoryless determined. On the other hand, as we discuss in Section 8, there is a cycle property Y which is closed under cyclic permutations and for which every arena $\mathcal{A}$ is Y-resettable, even though Y does not satisfy the condition. Thus, this condition cannot replace Y-resettability as a necessary condition for memoryless determinacy of all games in $\mathrm{Fcg}(Y)$. However, it is applicable in a wide variety of cases, and is usually very easy to check. Consider for example the cycle properties given in Section 2. For each of these properties Y, while proving that every arena is Y-resettable may not be hard, checking whether Y satisfies the condition above is almost completely trivial. Note that cyc-EvenLen, which is the only property among these which is closed under cyclic permutations but fails to satisfy this condition, would also fail to satisfy any other condition that guarantees memoryless determinacy since it actually admits FCGs that are not memoryless determined (cf. Theorem 3, Page 204).

Our goal in the rest of this section is to prove the following theorem:

Theorem 9 *(Easy to check). Let $Y \subseteq \mathbb{U}^*$ be a cycle property. If Y is closed under cyclic permutations, and both Y and $\neg Y$ are closed under concatenation, then every arena $\mathcal{A}$ is Y-resettable.*

By Theorem 6 we get:

Corollary 4. *Let $Y \subseteq \mathbb{U}^*$ be a cycle property. If Y is closed under cyclic permutations, and both Y and $\neg Y$ are closed under concatenation, then every game in $\mathrm{Fcg}(Y)$ is uniform memoryless determined.*

Remark 2. Consider the cycle property $Y = \text{cyc-GoodForEnergy}$ (recall from the definitions this means that either the energy level is positive, or it is zero and the largest priority occurring is even). Note that Y is closed under cyclic permutations, and both Y and $\neg Y$ are closed under concatenation. Conclude that every game in $\mathrm{Fcg}(Y)$ is pointwise-memoryless determined. This fact allows one to obtain a proof of Lemma 4 in [5] that no longer relies on the incorrect result from [4].

We begin by introducing a new kind of objective $O^{\mathcal{A}}_{\mathsf{tail}}(Y)$:

Definition 7. For an arena $\mathcal{A}$ and cycle property Y, define the objective $O^{\mathcal{A}}_{\mathsf{tail}}(Y)$ to consist of all plays π such that there is a suffix π' of π with the property that every cycle in $\mathit{cycles}(\pi')$ satisfies Y.

Note that this is not the same as saying that $\lambda(C) \in Y$ for all but finitely many cycles C in $\mathit{cycles}(\pi)$.[10]

Definition 8. An arena $\mathcal{A}$ is *Y-unambiguous* if $O^{\mathcal{A}}_{\mathsf{tail}}(Y) \cap O^{\mathcal{A}}_{\mathsf{tail}}(\neg Y) = \emptyset$.

In other words, $\mathcal{A}$ is Y-unambiguous if no play in $\mathcal{A}$ has two suffixes, π, π' such that every cycle in the cyclic decomposition of π is in Y, and every cycle in the cyclic decomposition of π' is not in Y.

In order to prove Theorem 9 we have to show that, for Y satisfying the assumptions of the theorem, for every arena $\mathcal{A}$, every Player $\sigma \in \{0,1\}$, and every node z in $\mathcal{A}$, we have that $\mathsf{WR}^{\sigma}(\mathcal{A}, O_{\mathsf{first}}(Y)) = \mathsf{WR}^{\sigma}(\mathcal{A}, O_{z\text{-}\mathsf{first}}(Y))$. The proof is in three parts:

Part 1. If Y is closed under cyclic permutations, and both Y and $\neg Y$ are closed under concatenation, then every arena $\mathcal{A}$ is Y-unambiguous.
Part 2. If $\mathcal{A}$ is Y-unambiguous then $\mathsf{WR}^{\sigma}(\mathcal{A}, O_{\mathsf{first}}(Y)) = \mathsf{WR}^{\sigma}(\mathcal{A}, O_{\mathsf{tail}}(Y))$.
Part 3. If $\mathcal{A}$ is Y-unambiguous then $\mathsf{WR}^{\sigma}(\mathcal{A}, O_{z\text{-}\mathsf{first}}(Y)) = \mathsf{WR}^{\sigma}(\mathcal{A}, O_{\mathsf{tail}}(Y))$.

Observe that parts 2 and 3 imply that if an arena $\mathcal{A}$ is Y-unambiguous then it is Y-resettable.

We first need a couple of definitions and a few easy lemmas. Say that Y is *closed under insertions* if it is closed under concatenation and: $ac \in Y$ and $b \in Y$ imply that $abc \in Y$, for all $a, b, c \in \mathbb{U}^*$. The *closure of Y under insertions*, is defined to be the smallest subset of $\mathbb{U}^*$ (with respect to set containment) that contains Y and is closed under insertions.

Lemma 8. *If Y is closed under cyclic permutations and under concatenation then Y is closed under insertions.*

Proof. Assume that $ac, b \in Y$. We have that $ac \in Y \Longrightarrow ca \in Y \Longrightarrow cab \in Y \Longrightarrow abc \in Y$. The middle implication is since Y is closed under concatenation, and the other two implications are since Y is closed under cyclic permutations. □

Fix an arena $\mathcal{A} = (V_0, V_1, E, \mathbb{U}, \lambda)$. Given a simple path $s \in E^*$, and a path $u \in E^*$ such that $\mathsf{end}(s) = \mathsf{start}(u)$, write $\mathit{labels}(s,u) = \{\lambda(C) \mid C \in \mathit{cycles}(s,u)\}$ for the set of the labels of the cycles output by CycDec(s,u).

Lemma 9. *Given an arena $\mathcal{A} = (V_0, V_1, E, \mathbb{U}, \lambda)$, let s, u be paths in $\mathcal{A}$ such that u is a cycle and $\mathsf{end}(s) = \mathsf{start}(u) = v$. We have that:*

1. *if $\mathit{stack}(s,u) = s$ then the insertion closure of $\mathit{labels}(s,u)$ contains $\lambda(u)$;*
2. *if v is the only node that appears on both s and u then $\mathit{stack}(s,u) = s$.*

Proof. Sketch. For the first item, note that the assumption $\mathit{stack}(s,u) = s$ implies that the cycles in $\mathit{cycles}(s,u)$ contain exactly the edges of u. Hence, it is not hard to see that $\lambda(u)$ is in the insertion closure of $\mathit{labels}(s,u)$. Intuitively, the algorithm output the cycles in $\mathit{cycles}(s,u)$ by "popping them out" of u. Thus, we can reverse this process and reconstruct u using insertion operations on the elements of $\mathit{cycles}(s,u)$.

Details. More formally, let $C^1, \ldots C^m$ be the elements of $\mathit{cycles}(s,u)$ in the order they were output. We iteratively construct a string $w \in E^*$ until we get $w = u$. We start with w being the empty string, and count down from m to 1. At step i in the count, we insert the cycle C^i into the correct position in w, as follows: let j be such that $u_j = C^i_0$ (i.e., the first edge of C^i), and set $\mathrm{w} := \mathrm{w}_1 \ldots \mathrm{w}_h \cdot C^i \cdot \mathrm{w}_{h+1} \ldots \mathrm{w}_{|\mathrm{w}|}$, where h is the maximal index such that all the edges $\mathrm{w}_1, \ldots, \mathrm{w}_h$ are in $\{u_1, \ldots, u_{j-1}\}$.

It is easy to see that when the construction ends w contains exactly the edges appearing in $C^1, \ldots C^m$ and thus, as noted before, exactly the edges of u. We claim that the edges are also ordered correctly (i.e., if $a < b$ then w_a appears in u before w_b) and thus, $\mathrm{w} = u$ as required. Assume by way of contradiction that the correct ordering in w was violated for the first time when C^i was inserted. Let u_j be the first edge of C^i and h be the position where C^i was inserted into w, as defined before. Recall that after the insertion the constructed string is $\mathrm{w}_1 \ldots \mathrm{w}_h \cdot C^i \cdot \mathrm{w}_{h+1} \ldots \mathrm{w}_{|\mathrm{w}|}$. Observe that since all edges of C_i are internally ordered correctly, and all of them correctly appear (by our choice of h) after $\mathrm{w}_1 \ldots \mathrm{w}_h$, the only possible violation is that some edge u_t in C^i incorrectly appears before the edge $\mathrm{w}_{h+1} = u_r$, i.e., that $j < r < t$. Also note that at the step where C^i was output, all edges above u_j that were on the stack were popped, and all edges up to u_t were already processed. Hence, it can not be that the edge u_r was output by the algorithm after C^i, which is a contradiction to the fact that u_r is already in w before the insertion of C^i. This completes the proof of the claim.

[10] For instance, consider the arena in Fig. 4, take Y to be cyc-EvenLen, and let $\pi := [(v_1,v_2)(v_2,v_1)(v_1,v_3)(v_3,v_2)(v_2,v_4)(v_4,v_1)]^{\omega}$. Note that i) decomposing the suffix π' starting with the second edge results in all cycles having odd length, and ii) it is not the case that almost all cycles in the cycles-decomposition of π (i.e., starting with the first edge) have odd length – in fact, they all have even length.

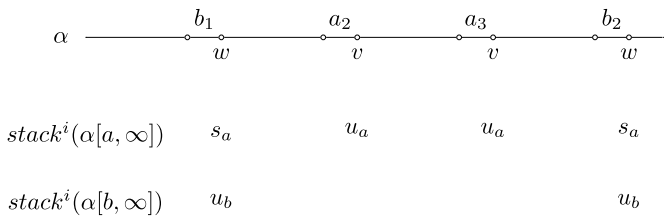

Fig. 6. Visualisation of the processing of the play α.

For the second item, observe that the assumption made there implies that, while processing u, the algorithm CycDec(s,u) cannot pop any edge in s. Hence, $stack(s,u)=s$ does not hold only if there exists j such that the edge $u_j=(v,v')$ is pushed on top of s but never popped. Observe that $v'\neq v=src(u)=trg(u)$ (the first inequality is since a self-loop is always popped, the rest by our assumption that u is a cycle that starts in v), and thus u_j is not the last edge of u. But this is a contradiction since the (non-empty) suffix $u_{j+1}\dots u_{|u|}$ of u contains at least one edge e' with $\mathsf{end}(e')=v$ (namely $u_{|u|}$), and the algorithm would have popped the edge u_j when it encountered the first edge that ends in v. □

We are now ready to show part 1 in the proof of Theorem 9.

Proposition 2. *Let $Y\subseteq\mathbb{U}^*$ be a cycle property. If Y is closed under cyclic permutations, and both Y and $\neg Y$ are closed under concatenation, then every arena $\mathcal{A}$ is Y-unambiguous.*

Proof. First, note that since Y is closed under cyclic permutations then so is $\neg Y$. By Lemma 8, we have that Y, as well as $\neg Y$, are closed under insertions.

Assume by way of contradiction that there is an arena $\mathcal{A}=(V_0,V_1,E,\mathbb{U},\lambda)$ which is not Y-unambiguous, and take a play α in the intersection of $O^{\mathcal{A}}_{\mathsf{tail}}(Y)$ and $O^{\mathcal{A}}_{\mathsf{tail}}(\neg Y)$. Let $a,b\in\mathbb{N}$ be such that every cycle in $cycles(\alpha[a,\infty])$ satisfies Y, and every cycle in $cycles(\alpha[b,\infty])$ satisfies $\neg Y$.

For i,j such that $a\leq i<j$, write $Out(i,j)=\{cycle^l(\alpha[a,\infty])\mid i\leq l\leq j\}\setminus\{\epsilon\}$ to be the set of cycles output by CycDec$(\epsilon,\alpha[a,\infty])$ while processing the edges $\alpha_i\dots\alpha_j$, and note that for every $C\in Out(i,j)$ we have that $\lambda(C)\in Y$. Similarly, for i,j such that $b\leq i<j$, write $Out^{\neg}(i,j)=\{cycle^l(\alpha[b,\infty])\mid i\leq l\leq j\}\setminus\{\epsilon\}$ to be the set of cycles output by CycDec$(\epsilon,\alpha[b,\infty])$ while processing the edges $\alpha_i\dots\alpha_j$, and note that for every $C\in Out^{\neg}(i,j)$ we have that $\lambda(C)\in\neg Y$.

Since, for every $i\geq 1$, the stack content $stack^i(\alpha[a,\infty])$ is of length at most $|V|-1$, there is at least one stack content that appears infinitely often. Thus, let $u_a\in E^*$ be a path of minimal length such that the set $A=\{i\in\mathbb{N}\mid u_a=stack^i(\alpha[a,\infty])\}$ is infinite. Similarly, let $u_b\in E^*$ be a path of minimal length such that $B=\{i\in\mathbb{N}\mid u_b=stack^i(\alpha[b,\infty])\}$ is infinite.

We assume w.l.o.g. (otherwise we simply replace A and B by appropriate infinite subsets of themselves until each of the following conditions is satisfied) that †:

(i) $u_a=stack^i(\alpha[a,\infty])$ for every $i\in A$, and $u_b=stack^i(\alpha[b,\infty])$ for every $i\in B$;
(ii) $\mathsf{end}(\alpha_i)=\mathsf{end}(\alpha_j)$ for all $i,j\in A$ and $\mathsf{end}(\alpha_i)=\mathsf{end}(\alpha_j)$ for all $i,j\in B$ (recall that the nodes on α come from the finite set V);
(iii) there exists $s_a\in E^*$ such that $s_a=stack^i(\alpha[a,\infty])$ for all $i\in B$.

Pick indices $b_1,b_2\in B$ and $a_2,a_3\in A$ in such a way that $b_1<a_2<a_3<b_2$. Fig. 6 may aid in visualising the current state of affairs. In this figure, $w:=\mathsf{end}(\alpha_{b_1})=\mathsf{end}(u_b)=\mathsf{end}(\alpha_{b_2})$ and $v:=\mathsf{end}(\alpha_{a_2})=\mathsf{end}(u_a)=\mathsf{end}(\alpha_{a_3})$.

Our aim now is to show that the above state of affairs leads to a contradiction, and thus deduce that it can not be that $\alpha\in O^{\mathcal{A}}_{\mathsf{tail}}(Y)\cap O^{\mathcal{A}}_{\mathsf{tail}}(\neg Y)$. The contradiction we will show is that $\lambda(\alpha[b_1+1,b_2])$ is both in Y and in its complement $\neg Y$, which is impossible. We will do that by showing how to build $\lambda(\alpha[b_1+1,b_2])$ in two different ways: the first by using cyclic permutations and concatenations of strings that are the labels of cycles that satisfy Y, and the second by doing the same but with cycles that satisfy $\neg Y$. Since by the assumption of the proposition Y and $\neg Y$ are closed under these string operations, the contradiction is reached.

We first show that $\alpha[b_1+1,b_2]$ (which by †(*ii*) is a cycle) satisfies $\neg Y$. Intuitively (refer to Fig. 6), this follows from the fact that all cycles output by the algorithm CycDec$(\epsilon,\alpha[b,\infty])$ while processing $\alpha[b_1+1,b_2]$ satisfy $\neg Y$, and the fact that before and after processing $\alpha[b_1+1,b_2]$ the stack of this algorithm is u_b, and thus by Lemma 9 we have that $\lambda(\alpha[b_1+1,b_2])$ is in the insertion closure of the labels of these cycles, and thus also in $\neg Y$ (recall that $\neg Y$ is closed under insertions).

We next show that $\lambda(\alpha[b_1+1,b_2])\in Y$. Observe (refer to Fig. 6) that $\alpha[b_1+1,b_2]=\alpha[b_1+1,a_2]\alpha[a_2+1,a_3])\alpha[a_3+1,b_2]$. Since Y is closed under concatenation and cyclic permutations, to show that $\lambda(\alpha[b_1+1,b_2])\in Y$ it is enough to show that $\lambda(\alpha[a_2+1,a_3])\in Y$ and $\lambda(x)\in Y$, where $x:=\alpha[a_3+1,b_2]\alpha[b_1+1,a_2]$. The fact that $\lambda(\alpha[a_2+1,a_3])\in Y$ follows from a symmetric argument that mimics the one used above to prove that $\alpha[b_1+1,b_2]$ satisfies $\neg Y$. It remains to show that $\lambda(x)\in Y$.

To see that $\lambda(x) \in Y$, note that when the algorithm CycDec$(\epsilon, \alpha[a, \infty])$ finishes processing $\alpha[a_3 + 1, b_2]$ it has the same stack content (namely s_a) that it had when it previously started processing $\alpha[b_1 + 1, a_2]$. Thus, if we imagine that after processing $\alpha[a_3 + 1, b_2]$, instead of continuing to process $\alpha[b_2 + 1, \infty]$, we feed the algorithm again with $\alpha[b_1 + 1, a_2]$ (i.e., we let it process the second part of x), we will get (by Lemma 2) that it will behave exactly the same as when it first processed $\alpha[b_1 + 1, a_2]$ as part of the prefix $\alpha[a, a_2]$. We thus obtain that while processing x this way starting with stack u_a, the algorithm ends with stack u_a and outputs only cycles that satisfy Y (recall that all cycles output by CycDec$(\epsilon, \alpha[a, \infty])$ do). Thus, by Lemma 9, we have that $\lambda(x)$ is in the insertion closure of the labels of cycles that satisfy Y, and thus also in Y (recall that Y is closed under insertions). □

The following Proposition deals with Part 2 in the proof of Theorem 9.

Proposition 3. *Fix a Y-unambiguous arena $\mathcal{A}$. Then for $\sigma \in \{0, 1\}$,*

$$WR^{\sigma}(\mathcal{A}, O_{first}(Y)) = WR^{\sigma}(\mathcal{A}, O_{tail}(Y)).$$

Proof. By Theorem 7 (item 1) it is sufficient to show that if $\mathcal{A}$ is Y-unambiguous then the game $(\mathcal{A}, O_{\text{tail}}(Y))$ is Y-greedy. So, fix a play π in $\mathcal{A}$. If every cycle in the cycles-decomposition of π satisfies Y then certainly $\pi \in O_{\text{tail}}(Y)$ (just take π itself as the required suffix). On the other hand, if every cycle in the cycles-decomposition of π satisfies $\neg Y$ then for the same reason $\pi \in O_{\text{tail}}(\neg Y)$. However, since $\mathcal{A}$ is Y-unambiguous, $\pi \notin O_{\text{tail}}(Y)$, as required. □

The following Proposition deals with Part 3.

Proposition 4. *Fix a Y-unambiguous arena $\mathcal{A}$ and vertex $z \in V$. Then for $\sigma \in \{0, 1\}$,*

$$WR^{\sigma}(\mathcal{A}, O_{z\text{-}first}(Y)) = WR^{\sigma}(\mathcal{A}, O_{tail}(Y)).$$

Proof. Let $\mathcal{A} = (V_0, V_1, E, \mathbb{U}, \lambda)$. We first claim that for every $\sigma \in \{0, 1\}$, we have that $\mathsf{WR}^{\sigma}(\mathcal{A}, O_{z\text{-first}}(Y)) \subseteq \mathsf{WR}^{\sigma}(\mathcal{A}, O_{\text{tail}}(Y))$. To see that the Proposition follows from this claim note that $\mathsf{WR}^0(\mathcal{A}, O_{z\text{-first}}(Y))$ and $\mathsf{WR}^1(\mathcal{A}, O_{z\text{-first}}(Y))$ partition V since the game $(\mathcal{A}, O_{z\text{-first}}(Y))$, being finitary, is determined (by Lemma 3). Since $\mathsf{WR}^0(\mathcal{A}, O_{\text{tail}}(Y))$ and $\mathsf{WR}^1(\mathcal{A}, O_{\text{tail}}(Y))$ are disjoint, the containments above must be equalities.

To prove the claim, recall that since $\mathcal{A}$ is Y-unambiguous then $O^{\mathcal{A}}_{\text{tail}}(Y) \cap O^{\mathcal{A}}_{\text{tail}}(\neg Y) = \emptyset$. Hence, a play π in the game $(\mathcal{A}, O_{\text{tail}}(Y))$ is won by Player 0 (resp. Player 1) if π has a suffix π' for which every cycle in $cycles(\pi')$ is in Y (resp. $\neg Y$). It is thus enough to show that: (†) for every $\sigma \in \{0, 1\}$, and every node v in $\mathcal{A}$, given a strategy S that is winning from v for Player σ in the game $(\mathcal{A}, O_{z\text{-first}}(Y))$, we can construct a strategy T for Player σ in the game $(\mathcal{A}, O_{\text{tail}}(Y))$, such that every play $\pi \in plays(T, v)$ has a suffix π' for which every cycle in $cycles(\pi')$ satisfies Y^{σ}.

Consider first the case where $z \notin reach(S, v)$, or that $z \in reach(S, v)$ but that every path in $plays(S, v)$ that visits z contains a cycle before the first occurrence of z on the path. Observe that this implies that S is winning from v in the game $(\mathcal{A}, O_{\text{first}}(Y))$. In this case we let $T = S^{\circlearrowleft}$, where $S^{\circlearrowleft}$ is the pumping strategy of S (Definition 6). By Lemma 7 we have that, for every $\pi \in plays(T, v)$, all the cycles in $cycles(\pi)$ satisfy Y^{σ}, and thus (†) holds with $\pi' = \pi$.

Consider now the remaining case that $z \in reach(S, v)$ and that there is a simple path ρ from v to z which is consistent with S. Define a strategy S_z for Player σ in the game $(\mathcal{A}, O_{\text{first}}(Y))$ as follows: for every $u \in V^*V_{\sigma}$, let $S_z(u) := S(\rho u)$ if u starts in z, and otherwise define it arbitrarily (i.e., $S_z(u) = w$, where w is some node with $(\mathsf{end}(u), w) \in E$). Observe that, for every $\pi \in plays(S_z, z)$, the play $\rho\pi$ is in $plays(S, v)$, and since ρ is a simple path that ends in z, Player σ wins the play $\rho\pi$ in the game $(\mathcal{A}, O_{z\text{-first}}(Y))$ iff the first cycle on π satisfies Y^{σ}. Thus, since S is winning from v, we have that S_z is winning from z in the game $(\mathcal{A}, O_{\text{first}}(Y))$.

We can now construct the desired strategy T for Player σ in the game $(\mathcal{A}, O_{\text{tail}}(Y))$. The strategy T works as follows: as long as a play does not touch the node z the strategy T behaves like the pumping strategy $S^{\circlearrowleft}$ of S; however, once (and if) z is reached, T erases its memory and switches to behave like the pumping strategy $(S_z)^{\circlearrowleft}$ of S_z (starting from z). Recall that we have to show that every play $\pi \in plays(T, v)$ has a suffix π' for which every cycle in $cycles(\pi')$ satisfies Y^{σ}. Informally, if z does not appear on π then π is consistent with $S^{\circlearrowleft}$, and by Lemma 7 every cycle in $cycles(\pi)$ satisfies Y^{σ}, and we can take $\pi' = \pi$. On the other hand, if z appears on π then $tail_z(\pi)$ is consistent with $(S_z)^{\circlearrowleft}$, and thus by Lemma 7 every cycle in $cycles(tail_z(\pi))$ satisfies Y^{σ}, and we can take $\pi' = tail_z(\pi)$.

Formally, define the strategy T as follows: for every $u \in V^*V_{\sigma}$,

$$T(u) := \begin{cases} S^{\circlearrowleft}(u) & \text{if } z \text{ does not appear on } u, \\ (S_z)^{\circlearrowleft}(tail_z(u)) & \text{otherwise.} \end{cases}$$

Given $\pi \in plays(T, v)$, either z appears on π or not. If it does not, then $\pi \in plays(S^{\circlearrowleft}, v)$, and by Lemma 7 every cycle C in $cycles(\pi)$ is the first cycle of some path ρ consistent with S and starting with v, that only uses edges from π. Thus, z does not appear on ρ, and since S is winning from v in the game $(\mathcal{A}, O_{z\text{-first}}(Y))$, we have that C satisfies Y^{σ}. If z does appear

on π, we argue that $tail_z(\pi) \in plays((S_z)^{\circlearrowleft}, z)$, i.e., that $tail_z(\pi)$ is consistent with $(S_z)^{\circlearrowleft}$. Indeed, if $m = |head(\pi)|$, then for every $j \geq 1$ for which $\mathsf{start}(tail_z(\pi)_j) \in V_\sigma$ we have:

$$\begin{aligned}\mathsf{end}(tail_z(\pi)_j) &= T(\pi[1, m+j]) \\ &= (S_z)^{\circlearrowleft}(tail_z(\pi[1, m+j])) \\ &= (S_z)^{\circlearrowleft}(\pi[m+1, m+j]) \\ &= (S_z)^{\circlearrowleft}((tail_z(\pi))[1, j]).\end{aligned}$$

By Lemma 7, since $tail_z(\pi) \in plays((S_z)^{\circlearrowleft}, z)$, every cycle in $cycles(tail_z(\pi))$ satisfies Y^σ. Overall, in the first case (†) holds with $\pi' = \pi$, and in the second with $\pi' = tail_z(\pi)$, which completes the proof of the claim. □

This concludes the proof of Theorem 9.

8. A recipe for proving that a game is memoryless determined

We synthesise the results of the previous section and provide a practical way of deducing uniform memoryless determinacy of many infinite-duration games.

First, we get the following sufficient condition for memoryless determinacy from Theorems 8 and 9.

Theorem 10. *Let Y be a cycle property that is closed under cyclic permutations, and such that both Y and $\neg Y$ are closed under concatenation. Let W be a winning condition. Every Y-greedy game $(\mathcal{A}, O(W))$ is uniform memoryless determined.*

Second, many winning conditions for infinite-duration games (for example parity and mean-payoff) are such that the outcome of a play does not depend on any finite prefix of the play, but only on some "infinite" property of the play. Such winning conditions are usually called *prefix-independent*. Formally:

Definition 9. Say that a winning condition $W \subseteq \mathbb{U}^\omega$ is *prefix-independent* if $c_1 c_2 \ldots \in W$ implies that the suffix $c_i c_{i+1} \ldots \in W$ for every $i \in \mathbb{N}$.

In practice, showing whether or not a given W is prefix-independent is usually very easy. The relevance of prefix-independence to our work is captured by the following theorem.

Theorem 11. *Let W be a prefix-independent winning condition, and let Y be a cycle-property that is closed under cyclic permutations. For every arena $\mathcal{A}$, if the game $(\mathcal{A}, O(W))$ is Y-greedy then it is uniform memoryless determined.*

Proof. By Theorem 8 it is sufficient to prove that every arena $\mathcal{A}$ is Y-resettable. By Propositions 3 and 4 it is thus sufficient to prove that $\mathcal{A}$ is Y-unambiguous. So, assume by way of contradiction there is a play π and indices $a, b \in \mathbb{N}$ be such that every cycle in $cycles(\pi[a, \infty])$ satisfies Y, and every cycle in $cycles(\pi[b, \infty])$ satisfies $\neg Y$. Since $(\mathcal{A}, O(W))$ is Y-greedy on $\mathcal{A}$ we get that, in the game $(\mathcal{A}, O(W))$, Player 0 wins $\pi[a, \infty]$, and Player 1 wins $\pi[b, \infty]$. But this is a contradiction to the assumption that W is prefix-independent. □

Given a winning condition W and a set of arenas of interest $\mathcal{C}$ (in many cases $\mathcal{C}$ is taken to be all arenas), Theorems 10 and 11 suggest the following recipe for proving that the game $(\mathcal{A}, O(W))$ is uniform memoryless determined for every $\mathcal{A} \in \mathcal{C}$.

Step 1. Finitise the winning condition $W \subseteq \mathbb{U}^\omega$ to get a cycle property $Y \subseteq \mathbb{U}^*$ that is closed under cyclic permutations.
Step 2. Show that the game $(\mathcal{A}, O(W))$ is Y-greedy for every $\mathcal{A} \in \mathcal{C}$.
Step 3. Show that either:
  (a) Both Y and $\neg Y$ are closed under concatenation; or that:
  (b) W is prefix independent.

We illustrate the use of the recipe with some examples.

Example 1. We will use the recipe to prove that every parity game is memoryless determined. For Step 1, a natural finitisation of the parity condition $W \subset \mathbb{Z}^\omega$ — which says that the largest priority occurring infinitely often is even — is the cycle property $\mathsf{cyc\text{-}Parity} \subset \mathbb{Z}^*$ which says that the largest priority occurring is even. This property is clearly closed under

cyclic permutations. For Step 2, it is easy to verify that every parity game is cyc-Parity-greedy, and for Step 3, it is immediate that both cyc-Parity and ¬cyc-Parity are closed under concatenation (as it happens, it is also easy to check that W is prefix-independent).

Example 2. We now consider a slightly more subtle application of the recipe. Consider the following game,[11] played on an arena $\mathcal{A}$ whose vertices are labelled[12] using the alphabet $\mathbb{U} = \{a, b\}$. The winning condition $W \subset \mathbb{U}^\omega$ consists of those infinite sequences that contain infinitely many a's and infinitely many b's. The natural finite version of W is the cycle property $Y \subset \mathbb{U}^*$ consisting of strings which contain at least one a and at least one b, and it is clearly closed under cyclic permutations. To see that W is Y-greedy on $\mathcal{A}$, observe that if all cycles in $cycles(\pi)$ satisfy Y then certainly $\pi \in W$. On the other hand, if all cycles in $cycles(\pi)$ satisfy $\neg Y$ then no edge that appears on such a cycle goes from a node labelled a to a node labelled b. Thus by Lemma 1 (Page 199) π does not satisfy W.

Unfortunately, $\neg Y$ is not closed under concatenation, which prevents us from applying Step 3(a). However, since W is clearly prefix-independent, we can apply Step 3(b) and conclude that $(\mathcal{A}, O(W))$ is memoryless determined.

Example 3. We conclude with a more sophisticated use of the recipe, applied to the initial credit problem of energy games. We show that either there is an initial credit with which Player 0 (the "energy" player) wins, or that for every initial credit Player 1 wins. In both cases, we show that the winner can use a memoryless strategy. A natural finitisation of the energy condition $W_r \subset \mathbb{Z}^\omega$ — which says that at every point along a play, the sum of the initial credit r and the labels of the edges already traversed is not negative — is the cycle property cyc-Energy $\subset \mathbb{Z}^*$ which says that the sum of the numbers is non-negative. This property is clearly closed under cyclic permutations. Given an arena $\mathcal{A}$, consider the game $(\mathcal{A}, O_{\text{all}}(\text{cyc-Energy}))$. We first claim that if the initial credit is at least $r = -t(|V| - 1)$, where t is the minimum amongst the negative weights of the arena, the winning regions of the energy game and the game $(\mathcal{A}, O_{\text{all}}(\text{cyc-Energy}))$ coincide and winning strategies transfer from the latter to the former. To see that, observe that a play won by Player 1 in $(\mathcal{A}, O_{\text{all}}(\text{cyc-Energy}))$ is also won by her in the energy game since the energy along the play tends to $-\infty$. On the other hand, let π be a play won by Player 0 in $(\mathcal{A}, O_{\text{all}}(\text{cyc-Energy}))$, and consider some prefix π' of it. Recall that, by Lemma 1, at most $|V| - 1$ edges are not on $cycles(\pi')$, and thus the energy level at the end of π' is at least the initial credit plus $t(|V| - 1)$. Hence, an initial credit of r suffices for π to be winning for Player 0 also in the energy game. It remains to show, using the recipe, that $(\mathcal{A}, O_{\text{all}}(\text{cyc-Energy}))$ is memoryless determined. As noted before (see Example 1 after Definition 5) every $(\mathcal{A}, O_{\text{all}}(Y))$ is Y-greedy, which completes Step 2. Since step 3a is immediate for cyc-Energy we are done.

9. Discussion and future work

The central algorithmic problem for a class of determined graph games is to decide, given an arena and a starting vertex, which of the players has a winning strategy. We have seen a PSPACE upper bound on the complexity of solving games in FCG(Y) (assuming Y is computable in PSPACE), and that there are very simple cycle properties Y for which the complexity is PSPACE-complete. What about other Ys such as cyc-Parity and cyc-MeanPayoff$_\nu$? Our Strategy Transfer result (Theorem 7) implies that the complexity of solving FCG(Y) is the same as that of solving Y-greedy games. For instance, since parity games are cyc-Parity-greedy, and the complexity of solving them is not known to be in PTIME — except on restricted classes of arenas, e.g., bounded DAG-width ([2]) and bounded trap-depth ([9]) — the same is true of the complexity of solving games from FCG(cyc-Parity).

On the other hand, since there are cycle properties Y such that some games in FCG(Y) are memoryless determined and some are not (for instance, take $Y =$ cyc-EvenLen), the following algorithmic problem naturally presents itself: what is the complexity of deciding, given (a finite description of) Y and an arena $\mathcal{A}$, whether or not the game $(\mathcal{A}, O_{\text{first}}(Y))$ is memoryless determined? We believe that this problem can be addressed using techniques developed in this paper.

Finally, this paper has dealt with qualitative games (i.e., a player either wins or loses). The exact nature of the theory of quantitative first-cycle games over general cycle properties Y is still to be explored. To what extent do our techniques generalise to quantitative games? or to stochastic ones?

Acknowledgments

We thank Erich Grädel for stimulating feedback which led to an improvement of the recipe in Section 8. We thank the referees for their useful suggestions and careful reading. Benjamin Aminof was supported by the Austrian National Research Network S11403-N23 (RiSE) of the Austrian Science Fund (FWF) and by the Vienna Science and Technology Fund (WWTF) through grant ICT12-059.

[11] This example was kindly brought to our attention by Erich Grädel, and it prompted us to enhance the recipe that was published in the preliminary work [1] to include Step 3(b).

[12] Recall Remark 1 that states that all the results in this paper also apply to vertex labelling.

References

[1] B. Aminof, S. Rubin, First cycle games, in: Proceedings 2nd International Workshop on Strategic Reasoning, SR 2014, in: EPTCS, vol. 146, 2014, pp. 83–90.
[2] D. Berwanger, A. Dawar, P. Hunter, S. Kreutzer, Dag-width and parity games, in: B. Durand, W. Thomas (Eds.), Symposium on Theoretical Aspects of Computer Science, STACS 2006, in: Lecture Notes in Computer Science, vol. 3884, Springer, 2006, pp. 524–536.
[3] A. Bianco, M. Faella, F. Mogavero, A. Murano, Exploring the boundary of half-positionality, Ann. Math. Artif. Intell. 62 (1–2) (2011) 55–77.
[4] H. Björklund, S. Sandberg, S.G. Vorobyov, Memoryless determinacy of parity and mean payoff games: a simple proof, Theor. Comput. Sci. 310 (1–3) (2004) 365–378.
[5] K. Chatterjee, L. Doyen, Energy parity games, Theor. Comput. Sci. 458 (2012) 49–60.
[6] A. Ehrenfeucht, J. Mycielski, Positional strategies for mean payoff games, Int. J. Game Theory 8 (2) (1979) 109–113.
[7] D. Gale, F.M. Stewart, Infinite games with perfect information, in: H. Kuhn, A. Tucker (Eds.), Contributions to the Theory of Games, Volume II, in: Annals of Mathematics Studies, vol. AM-28, Princeton University Press, 1953, pp. 245–266.
[8] H. Gimbert, W. Zielonka, Games where you can play optimally without any memory, in: M. Abadi, L. de Alfaro (Eds.), International Conference on Concurrency Theory, CONCUR 2005, in: Lecture Notes in Computer Science, vol. 3653, 2005, pp. 428–442.
[9] A. Grinshpun, P. Phalitnonkiat, S. Rubin, A. Tarfulea, Alternating traps in Muller and parity games, Theor. Comput. Sci. 521 (2014) 73–91.
[10] E. Kopczynski, Half-positional determinacy of infinite games, in: International Colloquium on Automata, Languages and Programming, ICALP 2006, 2006, pp. 336–347.
[11] M. Sipser, Introduction to the Theory of Computation, PWS Publishing Company, 1997.
[12] U. Zwick, M. Paterson, The complexity of mean payoff games on graphs, Theor. Comput. Sci. 158 (1&2) (1996) 343–359.