The memory of ω-regular and $\mathrm{BC}(\mathbf{\Sigma}^0_2)$ objectives

Antonio Casares ✉ ⌂ (iD)
University of Warsaw, Poland

Pierre Ohlmann ✉ ⌂ (iD)
CNRS, Laboratoire d'Informatique et des Systèmes (LIS), Marseille, France

──── **Abstract** ────

In the context of 2-player zero-sum infinite duration games played on (potentially infinite) graphs, the memory of an objective is the smallest integer k such that in any game won by Eve, she has a strategy with $\leq k$ states of memory. For ω-regular objectives, checking whether the memory equals a given number k was not known to be decidable. In this work, we focus on objectives in $\mathrm{BC}(\mathbf{\Sigma}^0_2)$, i.e. recognised by a potentially infinite deterministic parity automaton. We provide a class of automata that recognise objectives with memory $\leq k$, leading to the following results:

- for ω-regular objectives, the memory can be computed in **NP**;
- given two objectives W_1 and W_2 in $\mathrm{BC}(\mathbf{\Sigma}^0_2)$ and assuming W_1 is prefix-independent, the memory of $W_1 \cup W_2$ is at most the product of the memories of W_1 and W_2.

Our results also apply to chromatic memory, the variant where strategies can update their memory state only depending on which colour is seen.

1 Introduction

Context: Strategy complexity in infinite duration games

We study infinite duration games on graphs in which two players, called Eve and Adam, interact by moving a token along the edges of a (potentially infinite) edge-coloured directed graph. Each vertex belongs to one player, who chooses where to move next during a play. This interaction goes on for an infinite duration, producing an infinite path in the graph. The winner is determined according to a language of infinite sequences of colours W, called the objective of the game; Eve aims to produce a path coloured by a sequence in W, while Adam tries to prevent this. This model is widespread for its use in verification and synthesis [15].

In order to achieve their goal, players use strategies, which are representations of the course of all possible plays together with instructions on how to act in each scenario. In this work, we are interested in optimal strategies for Eve, that is, strategies that guarantee a victory whenever this is possible. More precisely, we are interested in the complexity of such strategies, or in other words, in the succinctness of the representation of the space of plays.

Positionality. The simplest strategies are those that assign in advance an outgoing edge to each vertex owned by Eve, and always play along this edge, disregarding all the other

52nd International Colloquium on Automata, Languages, and Programming (ICALP 2025).
Editors: Keren Censor-Hillel, Fabrizio Grandoni, Joel Ouaknine, and Gabriele Puppis; Article No. 140;
pp. 140:1–140:18

Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

features of the play. All the information required to implement such a strategy appears in the game graph itself. Objectives for which such strategies are sufficient to play optimally are called positional (or memoryless). Understanding positionality has been the object of a long line of research. The landmark results of Gimbert and Zielonka [22] and Colcombet and Niwinski [18] gave a good understanding of which objectives are bi-positional, i.e. positional for both players.

More recently, Ohlmann proposed to use universal graphs as a tool for studying positionality (taking Eve's point of view) [27]. This led to many advances in the study of positionality [2, 28], and most notably, a characterisation of positional ω-regular objectives by Casares and Ohlmann [10], together with a polynomial time decision procedure (and some other important corollaries, more discussion below).

Strategies with memory.   However, in many scenarios, playing optimally requires distinguishing different plays that end in the same vertex. A seminal result of Büchi and Landweber [7] states that in finite games where the objective is an ω-regular language, the winner has a winning strategy that can be implemented by a finite automaton processing the edges of the game; this result was later extended to infinite game graphs by Gurevich and Harrington [23]. Here, the states of the automaton are interpreted as memory states of the strategy, and a natural measure of the complexity of a strategy is the number of such states. More precisely, the memory of an objective W is the minimal k such that whenever Eve wins a game with objective W, she has a winning strategy with k states of memory. For ω-regular objectives, this is always finite [7, 23], while the case of positionality discussed above corresponds to memory $k = 1$.

Characterising the memory of objectives has been a recurrent research subject since the 1990s. Notable results include the characterisation for Muller objectives by Dziembowski, Jurdziński, and Walukiewicz [19], or for closed objectives by Colcombet, Fijalkow and Horn [16]. However, these are all rather restricted classes of ω-regular objectives. The problem of deciding the memory of ω-regular objectives has been raised in many occasions (see for instance [24, Sect. 9.2], [4, Sect. 4], [17, Conclusions], or [3]), but prior to this work, even computing the memory of open ω-regular objectives (sometimes called regular reachability objectives) was not known to be decidable.

Chromatic memory.   In the special case where the automata implementing strategies are only allowed to read the colours on the edges of the game graph, we speak of chromatic memory. In his PhD thesis, Kopczyński showed that, for prefix-independent ω-regular objectives and over finite game graphs, the chromatic memory can be computed in exponential time [24, Theorem 8.14]. Recently, it was shown that computing the chromatic memory of some restricted subclasses of ω-regular objectives is in fact **NP**-complete: for Muller objectives [8] and for topologically open or closed objectives [3]. However, the chromatic and the unconstrained memory of objectives may differ, even exponentially [8, 9].

Unions of objectives.   The driving question in Kopczyński's PhD thesis [24] is whether prefix-independent positional objectives are closed under union, which has become known as Kopczyński's conjecture. Recently, Kozachinskiy [25] disproved this conjecture, but only for positionality over finite game graphs, and using non-ω-regular objectives. In fact, the conjecture is now known to hold for ω-regular objectives [10] and $\mathbf{\Sigma_2^0}$ objectives [28]. Casares and Ohlmann proposed a generalisation of this conjecture from positional objectives to objectives requiring memory [12, Conjecture 7.1] (see also [24, Proposition 8.11]):

▶ **Conjecture 1** (Generalised Kopczyński's conjecture). *Let $W_1, W_2 \subseteq \Sigma^\omega$ be two prefix-independent objectives with memory k_1 and k_2, respectively. Then $W_1 \cup W_2$ has memory at most $k_1 \cdot k_2$.*

Using the characterisation of [19], it is not hard to verify that the conjecture holds for Muller objectives.

$\mathrm{BC}(\mathbf{\Sigma}_2^0)$ **languages.** The results in this work apply not only to ω-regular languages, but to the broader class of $\mathrm{BC}(\mathbf{\Sigma}_2^0)$ languages. These are boolean combinations of languages in $\mathbf{\Sigma}_2^0$ (countable unions of closed languages), or equivalently, recognised by deterministic parity automata with infinitely many states [31, Sect. 5]. This class includes typical non-ω-regular examples such as energy or mean-payoff objectives (and therefore also their boolean combinations with ω-regular ones) but also broader classes such as unambiguous ω-petri nets [20] and deterministic ω-Turing machines (Turing machines with a Muller condition).

Contributions

Our main contribution is a characterisation of $\mathrm{BC}(\mathbf{\Sigma}_2^0)$ objectives with memory $\leq k$, stated in Theorem 7. It captures both the memory and the chromatic memory of objectives over infinite game graphs. The characterisation is based on the notion of k-wise ε-completable automata, which are parity automata with states partitioned in k chains, where each chain is endowed with a tight hierarchical structure encoded in the ε-transitions of the automaton.

From this characterisation, we derive the following corollaries:

1. **Decidability in NP.** Given a deterministic parity automaton $\mathcal{A}$, the memory (resp. chromatic memory) of $L(\mathcal{A})$ can be computed in **NP**.
2. **Generalised Kopczyński's conjecture.** We establish (and strengthen) Conjecture 1 in the case of $\mathrm{BC}(\mathbf{\Sigma}_2^0)$ objectives: if W_1 and W_2 are $\mathrm{BC}(\mathbf{\Sigma}_2^0)$ objectives with memory k_1 and k_2, and one of them is prefix-increasing, then the memory of $W_1 \cup W_2$ is $\leq k_1 \cdot k_2$.

Main technical tools: Universal graphs and ε-completable automata. As mentioned above, Ohlmann proposed a characterisation of positionality by means of universal graphs [27]. In 2023, Casares and Ohlmann extended this characterisation to objectives with memory $\leq k$ by considering partially ordered universal graphs [12]. Until now, universal graphs have been mainly used to show that certain objectives have memory $\leq k$ (usually for $k = 1$); this is done by constructing a universal graph for the objective. One technical novelty of this work is to exploit both directions of the characterisation, as we also rely on the existence of universal graphs to obtain decidability results.

Our characterisation is based on the notion of k-wise ε-completable automata, which extends the key notion of [10] from positionality to finite memory.

Comparison with [10]. In 2024, Casares and Ohlmann characterised positional ω-regular objectives [10], establishing decidability of positionality in polynomial time, and settling Kopczyński's conjecture for ω-regular objectives. Although the current paper generalises the notions and some of the results from [10] to the case of memory, as well as potentially infinite automata, the proof techniques are significantly different: while [10] is based on intricate successive transformations of parity automata, ours is based on an extraction method in the infinite and manipulates ordinal numbers. Though somewhat less elementary, our proof is notably shorter, and probably easier to read.

When instantiated to the case of memory 1, we thus extend Kopczyński's conjecture from ω-regular objectives to positional $\mathrm{BC}(\boldsymbol{\Sigma}_2^0)$ objectives. This also gives an easier proof of decidability of positionality in polynomial time[1] for ω-regular objectives. However, some of the results of [10] are not recovered with our methods: the finite-to-infinite and 1-to-2-player lifts, as well as closure of positionality under addition of neutral letters.

2   Preliminaries

We let Σ be a countable alphabet[2] and $\varepsilon \notin \Sigma$ be a fresh symbol that should be interpreted as a neutral letter. Given a word $w \in (\Sigma \cup \{\varepsilon\})^\omega$ we write $\pi_\Sigma(w)$ for the (finite or infinite) word obtained by removing all ε's from w; we call $\pi_\Sigma(w)$ the projection of w on Σ. An objective is a set $W \subseteq \Sigma^\omega$. Given an objective $W \subseteq \Sigma^\omega$, we let W^ε denote $\pi_\Sigma^{-1}(W) \subseteq (\Sigma \cup \{\varepsilon\})^\omega$.

Throughout the paper, we use Von Neumann's notation for ordinals: λ denotes the set of ordinals $< \lambda$. This also applies to finite numbers, e.g. $k = \{0, \ldots, k-1\}$. As is standard, we also identify cardinals with their initial ordinals.

2.1   Graphs, games and memory

We introduce notions pertaining to games and strategy complexity, as they will be central in the statement of our results. Nevertheless, we note that all our technical proofs will use these definitions through Theorem 5 below, and will not explicitly use games.

Graphs.   A Σ-graph G is given by a set of vertices $V(G)$ and a set of coloured, directed edges $E(G) \subseteq V(G) \times \Sigma \times V(G)$. We write $v \xrightarrow{c} v'$ for edges (v, c, v'). A path is a sequence of edges with matching endpoints $(v_0 \xrightarrow{c_0} v_1)(v_1 \xrightarrow{c_0} v_2) \ldots$ which we write as $v_0 \xrightarrow{c_0} v_1 \xrightarrow{c_1} \ldots$. Paths can be empty, finite, or infinite, and have a label $c_0 c_1 \ldots$. Throughout the paper, graphs are implicitly assumed to be without dead-end: every vertex has an outgoing edge.

We say that a vertex v in a Σ-graph (resp. a $(\Sigma \cup \{\varepsilon\})$-graph) satisfies an objective $W \subseteq \Sigma^\omega$ if the label of any infinite path from v belongs to W (resp. to W^ε). A pointed graph is a graph with an identified initial vertex. A pointed graph satisfies an objective $W \subseteq \Sigma^\omega$ if the initial vertex satisfies W; a non-pointed graph satisfies an objective if all its vertices do. An infinite tree is a sinkless pointed graph whose initial vertex is called the root, and with the property that every vertex admits a unique path from the root.

A morphism from a Σ-graph G to a Σ-graph H is a map $\phi \colon V(G) \to V(H)$ such that for any edge $v \xrightarrow{c} v'$ in G, it holds that $\phi(v) \xrightarrow{c} \phi(v')$ is an edge in H. Morphisms between pointed graphs should moreover send the initial vertex of G to the initial vertex of H. Morphisms need not be injective. We write $G \to H$ when there exists a morphism $\phi \colon G \to H$.

Games and strategies.   A game is given by a pointed $(\Sigma \cup \{\varepsilon\})$-graph G together with an objective $W \subseteq \Sigma^\omega$, and a partition of the vertex set $V(G) = V_{\mathrm{Eve}} \sqcup V_{\mathrm{Adam}}$ into the vertices controlled by Eve and those controlled by Adam. A strategy[3] (for Eve) is a pointed graph together with a morphism π towards G, satisfying that for every edge $v \xrightarrow{c} v'$ in the game,

---

[1]   It is explained in [10, Theorem 5.3] how decidability of positionality in polynomial time can be derived, with reasonable efforts, from Kopczyński's conjecture.

[2]   We restrict our study to countable alphabets, as if Σ is uncountable, the topological space Σ^ω is not Polish and the class $\mathrm{BC}(\boldsymbol{\Sigma}_2^0)$ is not as well-behaved.

[3]   We follow the terminology from [12]. The classical notion of a strategy as a function $f \colon E(G)^* \to V(G)$ can be recovered by considering the graph with vertices $E(G)^*$, and edges $\rho \xrightarrow{e} \rho e$.

where $v \in V_{\text{Adam}}$, and for all $u \in \pi^{-1}(v)$, there is an edge $u \xrightarrow{c} u'$ such that $u' \in \pi^{-1}(v')$. A strategy is winning if it satisfies the objective W of the game. We say that Eve wins if there exists a winning strategy.

Memory. A finite-memory strategy[4] is a strategy with vertices $V(G) \times k$ and projection $\pi(v, m) = v$, with the additional requirement that for every edge $(v, m) \xrightarrow{\varepsilon} (v', m')$, it holds that $m = m'$. We say that k is the memory of the strategy, and numbers $1, \dots, k$ are called memory states. Informally, the requirement above says that when reading an ε-transition in the game, we are not allowed to change the memory state; this is called ε-memory in [12] (to which we refer for more discussion), but since it is the main kind of memory in this paper, we will simply call it the memory.

A finite-memory strategy is called chromatic if there is a map $\chi : k \times (\Sigma \cup \{\varepsilon\}) \to k$ such that for every edge $(v, m) \xrightarrow{c} (v', m')$ in the strategy, it holds that $m' = \chi(m, c)$. We say that χ is the chromatic update. Note that necessarily, we have $\chi(m, \varepsilon) = m$ for every memory state m.

The (chromatic) memory of an objective W is the minimal k such that for every game with objective W, if Eve has a winning strategy, she has a winning (chromatic) strategy with memory $\leq k$.

2.2 Automata

We write d for the set $\{0, \dots, d\}$. A parity automaton $\mathcal{A}$ (or just automaton in the following) with index d – an even number – and alphabet Σ, is a pointed $((\Sigma \cup \{\varepsilon\}) \times d)$-graph. Vertices are called states, edges are called transitions and written $q \xrightarrow{c:y} q'$, where $c \in \Sigma \cup \{\varepsilon\}$ and $y \in d$. Elements in d are called priorities. Generally, we use the convention that even priorities are denoted with letter x, whereas y can be used to denote any priority. Transitions of the form $q \xrightarrow{\varepsilon:y} q'$ are called ε-transitions; note that they also carry priorities.

Infinite paths from the initial state q_0 are called runs. A run is accepting if the projection of its label on the second coordinate belongs to

$$\text{Parity}_d = \{y_0 y_1 \cdots \in d^\omega \mid \liminf(y) \text{ is even}\}. \quad \text{(Note the use of } min\text{-parity.)}$$

The language $L(\mathcal{A})$ of $\mathcal{A}$ is $\pi_\Sigma(L')$, where $L' \subseteq (\Sigma \cup \{\varepsilon\})^\omega$ is the set of projections on the first coordinate of runs which are accepting. We require that all these projections are infinite words; stated differently, there is no accepting run from q_0 labelled by a word in $(\Sigma \cup \{\varepsilon\})^* \varepsilon^\omega$. An automaton is deterministic if there are no ε-transitions and for any state $q \in V(\mathcal{A})$ and any letter $a \in \Sigma$ there is at most one transition $q \xrightarrow{a:} \_$. We say that an automaton is determinisable by pruning if one can make it deterministic by removing some transitions, without modifying its language. A language belongs to $\text{BC}(\mathbf{\Sigma}_2^0)$ if it is the language of a deterministic automaton, and it is ω-regular if the automaton is moreover finite.

We will often identify pointed graphs with automata of index 0, by labelling all transitions with priority 0. Note that in this case, all runs are accepting. This requires making sure that there is no accepting run labelled by a word from $\Sigma^* \varepsilon^\omega$, which, up to assuming that all vertices are accessible from the initial one, amounts to saying that there is no infinite path of $\xrightarrow{\varepsilon}$. We say that such a graph is well-founded.

---

[4] It is common to define a memory structure as an automaton reading the edges of a game graph. This notion can be recovered by taking k as the states of the automaton.

Blowups and k-automata.   A k-blowup $\mathcal{B}$ of an automaton $\mathcal{A}$ is any automaton with $V(\mathcal{B}) \subseteq V(\mathcal{A}) \times \{1, \ldots, k\}$, with initial state in $\{q_0\} \times k$ and such that for each transition $q \xrightarrow{a:y} q'$ in $\mathcal{A}$ and each $m \in k$, there is a transition $(q, m) \xrightarrow{a:y} (q', \_)$. We also allow for extra transitions in $\mathcal{B}$. Note that in this case $L(\mathcal{A}) \subseteq L(\mathcal{B})$.

A k-automaton is just an automaton whose states are a subset of $Q \times k$, for some set Q; for instance, k-blowups are k-automata. Equivalently, these are automata with a partition of their states in k subsets. For a state (q, m) in a k-automaton, m is called its memory state. A k-automaton is called chromatic if there is a map $\chi : \{1, \ldots, k\} \times (\Sigma \cup \{\varepsilon\}) \to \{1, \ldots, k\}$ such that for all transition $(q, m) \xrightarrow{a:y} (q', m')$ it holds that $m' = \chi(m, c)$.

Cascade products.   Let $\mathcal{A}$ be an automaton with alphabet Σ and index d, and let S be a d-graph. We define their cascade product $\mathcal{A} \ltimes S$ to be the $(\Sigma \cup \{\varepsilon\})$-graph with vertices $V(\mathcal{A}) \times V(S)$ and edges

$$(q, s) \xrightarrow{c} (q', s') \quad \Longleftrightarrow \quad \exists y, [q \xrightarrow{c:y} q' \text{ and } s \xrightarrow{y} s'].$$

If S is a pointed graph with intial vertex s_0, then $\mathcal{A} \ltimes S$ is pointed with initial vertex (q_0, s_0). It is easy to check that we then have the following lemma.

▶ **Lemma 2.** *Let $\mathcal{A}$ be an automaton with index d and S be a d-graph satisfying Parity_d. Then $\mathcal{A} \ltimes S$ is well-founded and satisfies $L(\mathcal{A})$.*

2.3 ε-completability and universal graphs

We now introduce and discuss the key notion used in our main characterisation, which adapts the notion from [10] from positionality to finite memory.

k-wise ε-completability.   A k-automaton $\mathcal{A}$ with index d is called k-wise ε-complete if for each even $x \in d$, for each memory state m and each ordered pair of states $(q, m), (q', m)$:

either $(q, m) \xrightarrow{\varepsilon:x} (q', m)$ or $(q', m) \xrightarrow{\varepsilon:x+1} (q, m)$.

Intuitively, having an edge $(q, m) \xrightarrow{\varepsilon:x} (q', m)$ means that "(q, m) is much better than (q', m)", as one may freely go from (q, m) to (q', m) and even see a good priority on the way. Similarly, $(q', m) \xrightarrow{\varepsilon:x+1} (q, m)$ means that "(q', m) is not much worse than (q, m)".

It is also useful for the intuition to apply the definition to both ordered pairs $((q, m), (q', m))$ and $((q', m), (q, m))$. Since automata exclude accepting runs which are ultimately comprised of ε-transitions, we cannot have $(q, m) \xleftrightarrow{\varepsilon:x} (q', m)$, and therefore ε-completability rewrites as: for each x, each memory state m and each unordered pair $(q, m), (q', m)$ of states,

either $(q, m) \xrightarrow[\varepsilon:x+1]{\varepsilon:x} (q', m)$ or $(q, m) \xleftarrow{\varepsilon:x+1} (q', m)$.

Hence an alternative, maybe more useful view (this is the point of view adopted in [10]) is that, up to applying some adequate closure properties, a k-wise ε-complete automaton is endowed with the following structure: for each even priority x and each memory state m, the states with memory state m are totally preordered by the relation $\xrightarrow{x+1}$, and the relation $\xrightarrow{x}$ is the strict version of this preorder. Moreover, for $x' > x$, the x'-preorder is a refinement of the x-preorder.

A k-automaton $\mathcal{A}$ is called k-wise ε-completable if one may add ε-transitions to it so as to turn it into a k-wise ε-complete automaton $\mathcal{A}^\varepsilon$ satisfying $L(\mathcal{A}^\varepsilon) = L(\mathcal{A})$. We simply say "$\varepsilon$-complete" (resp. "$\varepsilon$-completable") as a shorthand for "1-wise ε-complete" (resp. "1-wise" ε-completable).

▶ **Example 3.** Let $\Sigma = \{a, b, c\}$ and

$$W = \texttt{No}(b) \vee \texttt{Fin}(aa) \vee \texttt{Inf}(cc).$$

We show a 2-wise ε-complete automaton recognising W in Figure 1. By Theorem 7, the memory of W is ≤ 2 (and it is easy to see that this bound is tight).
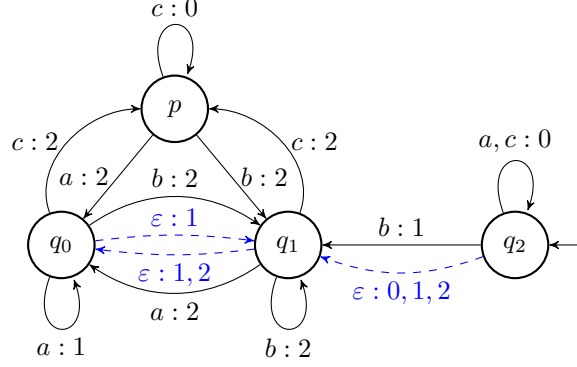
■ **Figure 1** A 2-automaton recognising $W = \texttt{No}(b) \vee \texttt{Fin}(aa) \vee \texttt{Inf}(cc)$, where p is assumed to have a different memory state than q_0, q_1 and q_2. It is 2-wise ε-completable by adding the indicated ε-transitions. A completion also contains the transitions $q_2 \xrightarrow{\varepsilon:0,1,2} q_0$, omitted for ease of reading. However, it is not chromatic since reading c may or may not switch the memory state.

The following theorem, a key result in [12] where it is called the structuration lemma, will also be crucial to this work. Recall that we see well-founded pointed graphs as automata with only 0-transitions, and we apply the terms k-blowup and ε-completable to them accordingly.

▶ **Theorem 4** (Adapted from Lemma 3.4 in [12]). *Let G be a well-founded pointed graph satisfying an objective W which is assumed to have (chromatic) memory $\leq k$ over games of size $\leq 2^{|G|}$. There is a (chromatic) k-blowup G' of G which is well-founded, k-wise ε-complete, and satisfies W.*

Universal graphs. Given an objective W and a cardinal κ, we say that a graph U is (κ, W)-universal if for any infinite tree T of cardinality $|V(T)| < \kappa$ satisfying W, there is a morphism $\phi : T \to U$ such that $\phi(t_0)$ satifies W in U, where t_0 is the root of T. We may now rephrase the main theorem of [12] in terms of ε-complete universal graphs.

▶ **Theorem 5** (Theorem 3.1 in [12]). *Let W be an objective. Then W has (chromatic) memory $\leq k$ if and only if for every cardinal κ there exists a (κ, W)-universal graph which is (chromatic and) k-wise ε-complete.*

We now give an explicit definition of a $(\kappa, \mathrm{Parity}_d)$-universal graph S_d^κ which is ε-complete. These ideas date back to the works of Streett and Emerson, who coined the name signatures [30], and were made more explicit by Walukiewicz [33]. Vertices are tuples of ordinals $< \kappa$, indexed by odd priorities in d and ordered lexicographically (with the smaller indices being the most significant). For a tuple s and index y, we let $s_{<y}$ be the tuple obtained from s by restricting to coordinates with index $< y$. Edges are given by

$$s \xrightarrow{y} s' \quad \Longleftrightarrow \quad \begin{cases} s_{<y} \geq s'_{<y} \text{ and } y \text{ is even; or} \\ s_{\leq y} > s'_{\leq y} \text{ and } y \text{ is odd,} \end{cases}$$

and ε-edges are given by $s \xrightarrow{\varepsilon} s'$ if and only if $s \geq s'$.

▶ **Lemma 6** ([11, Lemma 2.7]). *The graph S_d^κ is $(\kappa, \mathrm{Parity}_d)$-universal.*

We will work extensively with such tuples, as well as their prefixes and suffixes. For readability, we also use subscripts to indicate which (odd) coordinates are concerned, for instance $s_{<x}$ will be our notation for tuples of ordinals $< \kappa$ indexed with odd priorities $< x$, and similarly for $s_{>x}$. Concatenation of tuples is written like for words, therefore $s_{<x}s_{>x}$ denotes a tuple indexed by all odd priorities (i.e. an vertex of S_κ^d). We treat $s_{<x}$ and $s_{>x}$ as different variables, which we may quantify independently.

3   Characterisation of objectives in $\mathrm{BC}(\Sigma_2^0)$ with memory $\leq k$

We state our main characterisation theorem and its decidability consequences for ω-regular languages. We assume that the alphabet Σ is countable, therefore automata can also be taken with countable sets of states.

▶ **Theorem 7** (Main characterisation). *Let W be a $\mathrm{BC}(\Sigma_2^0)$ objective and let $k \in \mathbb{N}$. The following are equivalent:*

 (i.) *W has memory $\leq k$ (resp. chromatic memory $\leq k$) on games of size $\leq 2^{2^{\aleph_0}}$.*

 (ii.) *For any automaton $\mathcal{A}$ recognising W, there is a (chromatic) k-blowup $\mathcal{B}$ of $\mathcal{A}$ which is k-wise ε-complete and recognises W.*

 (iii.) *There is a deterministic (chromatic) k-automaton $\mathcal{A}$ which is k-wise ε-completable and recognises W. If W is recognised by a deterministic automaton of size n, then $\mathcal{A}$ can be taken of size kn.*

 (iv.) *For every cardinal κ, there is a (chromatic) (κ, W)-universal graph which is well-founded and k-wise ε-complete.*

 (v.) *W has (chromatic) memory $\leq k$ on arbitrary games.*

For ω-regular W given by a deterministic automaton $\mathcal{B}$ of size n, this allows to compute the (chromatic) memory in **NP**. First, we note that the memory of $L(\mathcal{B})$ is at most n, as the automaton itself can serve as a (chromatic) memory. Therefore, we can guess $k \leq n$, a deterministic automaton $\mathcal{A}$ of size $\leq kn$ and a (chromatic) k-wise ε-completion $\mathcal{A}^\varepsilon$, and check if $L(\mathcal{B}) \subseteq L(\mathcal{A})$ and if $L(\mathcal{A}^\varepsilon) \subseteq L(\mathcal{B})$, which can be done in polynomial time, since $\mathcal{A}$ and $\mathcal{B}$ are deterministic [14]. Prior to our work, neither computing the memory nor the chromatic memory was known to be decidable (although the chromatic memory over finite graphs can be computed in exponential time [24]).

▶ **Corollary 8** (Decidability in **NP**). *Given an integer k and a deterministic automaton $\mathcal{A}$, the problem of deciding if $L(\mathcal{A})$ has (chromatic) memory $\leq k$ belongs to **NP**.*

Our main contribution is the implication from (i) to (ii), which is the object of Section 3.1. We proceed in Section 3.2 to show that (ii) implies (iii) which is straightforward. The implication (iii) $\implies$ (iv) is adapted from [10] and presented in Section 3.3. Finally, the implication (iv) $\implies$ (v) is the result of [12] (Theorem 5), and the remaining one is trivial.

3.1   Existence of k-wise ε-complete automata: Proof overview

We start with the more challenging and innovative implication: how to obtain a k-wise ε-complete automaton given an objective in $\mathrm{BC}(\Sigma_2^0)$ with memory k (that is, (i) $\implies$ (ii)). In this Section, we give a detailed overview of the proof.

▶ **Proposition 9.** *Let W be an objective recognised by an automaton $\mathcal{A}$, and assume that W has (chromatic) memory $\leq k$ on games of size $\leq 2^{2^{\aleph_0}}$. Then there is a (chromatic) k-blowup $\mathcal{B}$ of $\mathcal{A}$ recognising W which is k-wise ε-complete.*

We assume that W has memory $\leq k$ on games of size $\leq \kappa$; we discuss the chromatic case at the end of the section. Let $\mathcal{A}$ be an automaton recognising W; we aim to construct a k-blowup $\mathcal{B}$ of $\mathcal{A}$ which is ε-complete. We let S denote S_d^κ, the $(\kappa, \mathrm{Parity}_d)$-universal graph defined above.

We consider the cascade product $\mathcal{A} \ltimes S$; this is a $(\Sigma \cup \{\varepsilon\})$-graph which intuitively encodes all possible accepting behaviours in $\mathcal{A}$. Then we apply the structuration result (Theorem 4) to $\mathcal{A} \ltimes S$ which yields a k-blowup G of $\mathcal{A} \ltimes S$ which is well-founded and k-wise ε-complete (as a graph). Stated differently, up to blowing the graph $\mathcal{A} \ltimes S$ into k copies, we have been able to endow it with many ε-transitions, so that over each copy, $\xrightarrow{\varepsilon}$ defines a well-order. Note that the states of G are of the form (q, m, s), with $q \in V(\mathcal{A})$, $m \in$ and $s \in V(S)$.

The states of $\mathcal{B}$ will be $V(\mathcal{B}) = V(\mathcal{A}) \times k$. The challenge lies in defining the transitions in $\mathcal{B}$, based on those of G.

Given a state $(q, m) \in V(\mathcal{B})$ and a transition $q \xrightarrow{c:y} q'$ in $\mathcal{A}$, where $c \in \Sigma \cup \{\varepsilon\}$, by applying the definitions we get transitions of the form $(q, m, s) \xrightarrow{c} (q', m', s')$ in G, for different values of m', whenever $s \xrightarrow{y} s'$ in S. We will therefore define transition $(q, m) \xrightarrow{c:y} (q', m')$ in $\mathcal{B}^\varepsilon$ if m' matches suitably many transitions as above; for now, we postpone the precise definition.

We should then verify that the obtained automaton $\mathcal{B}$:

- is a k-blowup of $\mathcal{A}$,
- is ε-complete, and
- recognises W.

The first two items above state that $\mathcal{B}$ should have many transitions: at least those inherited from $\mathcal{A}$, and in addition a number of ε-transitions. This creates a tension with the third item, which states that even with all these added transitions, the automaton $\mathcal{B}$ should not accept too many words.

Let us focus on the third item for now, which will lead to a correct definition for $\mathcal{B}$. Take an accepting run

$$(q_0, m_0) \xrightarrow{c_0:y_0} (q_1, m_1) \xrightarrow{c_1, y_1} \dots$$

in $\mathcal{B}$, where $x = \liminf_i y_i$ is even; for the sake of simplicity, assume that all y_i's are $\geq x$. We should show that its labelling $w = c_0 c_1 \dots$ belongs to W. To this end, we will decorate the run with labels $s_0, s_1, \dots \in S$ so that

$$(q_0, m_0, s_0) \xrightarrow{c_0} (q_1, m_1, s_1) \xrightarrow{c_1} \dots$$

defines a path in G, which concludes since G satisfies W.

Recall that the elements of S are tuples of ordinals $< \kappa$ indexed by $d/2$ odd priorities. We use $s_{<x}$ (resp. $s_{>x}$) to refer to a tuple indexed by priorities up to $x - 1$ (resp. from $x + 1$). We also let $x_{\mathrm{odd}} = \{1, 3, \dots, x - 1\}$.

To construct the s_i's, we fix a well chosen prefix $s_{<x} \in \kappa^{x_{\mathrm{odd}}}$ which will be constant, and proceed as follows.

(a.) If $y_i = x$, then we set $s_i = s_{<x} s_{>x}$, for some $s_{>x}$ which depends only on c_i.
(b.) If $y_i < x$, then we set $s_i = s_{<x} s_{>x}$, for some $s_{>x}$ which depends on c_i as well as s_{i+1}.

At this stage the reader may be worried that the backward induction underlying the above definition is not well-founded; however, since the first case occurs infinitely often,
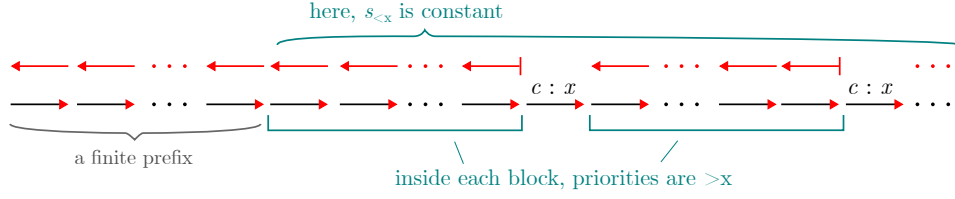
Figure 2 The run in $\mathcal{B}$ (in black), and the order in which the s_i's are computed (in red).

the backward induction from the second item is only performed over finite blocks, see also Figure 2.

This leads to the following definition for $\mathcal{B}$, where x is an even priority:

$$(q, m) \xrightarrow{c:x} (q', m') \text{ in } \mathcal{B} \iff \exists s_{<x} \exists s_{>x} \forall s'_{>x}, (q, m, s_{<x} s_{>x}) \xrightarrow{c} (q', m', s_{<x} s'_{>x}) \text{ in } G,$$
$$(q, m) \xrightarrow{c:x+1} (q', m') \text{ in } \mathcal{B} \iff \exists s_{<x} \forall s'_{>x} \exists s_{>x}, (q, m, s_{<x} s_{>x}) \xrightarrow{c} (q', m', s_{<x} s'_{>x}) \text{ in } G.$$

The first line corresponds to point (a.) above, where $s_{>x}$ can be chosen independently of $s'_{>x}$, whereas the second line corresponds to point (b.), since the choice of $s_{>x}$ is conditioned on the value of $s'_{>x}$. (For priorities $> x + 1$, we may apply either the first or second line, depending on the parity, to get the required conclusion.)

The remaining issue is that the choice of the fixed common prefix $s_{<x}$ should be made uniformly, regardless of the transition. This is achieved thanks to an adequate extraction lemma (which extends the pigeonhole principle to the case at hands), which finds a large enough subset T of $\kappa^{[d]_{\mathrm{odd}}}$, so that transitions $(q, m, s) \to (q', m', s')$ are similar for different choices of $s, s' \in T$. This ensures that $s_{<x}$ can be chosen uniformly.

There remains to verify that $\mathcal{B}$ is indeed a k-blowup of $\mathcal{A}$ and that it is ε-complete, which will follow easily from the definitions and ε-completeness of G (this is because, after removing "$\exists s_{<x}$" from the definition above, the second line resemble the negation of the first).

For the chromatic case, the proof is exactly the same, we should simply check that if G is chromatic (which is guaranteed by Theorem 4), then the so is the obtained automaton $\mathcal{B}$.

3.2 Existence of deterministic ε-completable automata

We now prove the implication from (ii) to (iii) in Theorem 7. Take $\mathcal{A}$ to be a deterministic automaton recognising W, and let $\mathcal{B}^\varepsilon$ be the obtained k-blowup of $\mathcal{A}$ which is (chromatically) k-wise ε-complete and recognises W. Then let $\mathcal{B}$ be obtained from $\mathcal{B}^\varepsilon$ by only keeping, for each state $(q, m) \in V(\mathcal{B})$ and each transition $q \xrightarrow{a:y} q'$ in $\mathcal{A}$, a single transition of the form $(q, m) \xrightarrow{a:y} (q', m')$ chosen arbitrarily. Note that $\mathcal{B}$ is a k-blowup of $\mathcal{A}$, so we have: $L(\mathcal{A}) \subseteq L(\mathcal{B}) \subseteq L(\mathcal{B}^\varepsilon)$. We conclude that $\mathcal{B}$ is a deterministic (chromatically) k-wise ε-completable automaton recognising W, as required.

3.3 From deterministic ε-completable automata to universal graphs

We now prove the implication from (iii) to (iv) in Theorem 7. This result was already proved in [11, Prop. 5.30] for the case of $k = 1$; extending to greater values for k presents no difficulty.

Let $\mathcal{B}$ be a deterministic[5] (chromatic) k-wise ε-completable automaton recognising W

---

[5] For the purpose of this proof, history-determinism would be sufficient (we refer to [1] for the definition

and $\mathcal{B}^\varepsilon$ be an ε-completion. Fix a cardinal κ and let S denote S_d^κ. Consider the cascade product $U = \mathcal{B}^\varepsilon \ltimes S$. We claim that U is (chromatic) k-wise ε-complete and (κ, W)-universal. Universality follows from the facts that $\mathcal{B}$ is deterministic and S is $(\kappa, \mathrm{Parity}_d)$-universal.

$\triangleright$ **Claim 10.** The graph $U = \mathcal{B}^\varepsilon \ltimes S$ is (κ, W)-universal.

Proof. Take a tree T of size $< \kappa$ which satisfies W. We should show that $T \to U$. Since $\mathcal{B}$ is deterministic, we can define a labelling $\rho \colon V(T) \to V(\mathcal{B})$ by mapping t_0 to q_0 and $t \mapsto q$ if the run of $\mathcal{B}$ on the finite word labelling the path $t_0 \to t$ ends in q. Then, any infinite path from t_0 in T is mapped to a run in $\mathcal{B}$ that is accepting (since T satisfies W). Therefore the tree T' obtained by taking T and replacing edge-labels by the priorities appearing in their ρ-images satisfies Parity_d and has size $< \kappa$, so there is a morphism $\mu \colon T \to S$. It is a direct check that $(\rho, \mu) \colon V(T) \to V(\mathcal{B}) \times V(S) = V(U)$ indeed defines a morphism. $\triangleleft$

Showing that U is k-wise ε-complete is slightly trickier.

$\triangleright$ **Claim 11.** The graph $U = \mathcal{B}^\varepsilon \ltimes S$ is well-founded and (chromatic) k-wise ε-complete.

Proof. Well-foundedness of U follows directly from Lemma 2. Let us write $B_1, \ldots, B_k$ for the k parts of $\mathcal{B}^\varepsilon$; the k parts of U will be $B_1 \times V(S), \ldots, B_k \times V(S)$. If applicable, chromaticity is a direct check. Let $(b, s), (b', s') \in V(U)$ be in the same part, i.e. $b, b' \in B_i$ for some i. Let x_0 be the minimal even priority such that $b \xrightarrow{\varepsilon : x_0} b'$ in $\mathcal{B}^\varepsilon$ (if such an x does not exist then $x_0 = d + 2$). Then let y_0 be the minimal odd priority such that $s'_{\leq y_0} > s_{\leq y_0}$ (as previously, if $s = s'$ then we let $y_0 = d + 1$). We distinguish two cases.
(1) If $x_0 < y_0$. Then we have $b \xrightarrow{\varepsilon : x_0} b'$ in $\mathcal{B}^\varepsilon$ and $s_{<x_0} = s'_{<x_0}$ which gives $s \xrightarrow{x_0} s'$ in S. Thus we get $(b, s) \xrightarrow{\varepsilon} (b', s')$ in U.
(2) If $y_0 < x_0$. Then $s'_{\leq y_0} > s_{\leq y_0}$, which gives $s' \xrightarrow{y_0} s$ in S. Since $\mathcal{B}^\varepsilon$ is k-wise ε-complete and by definition of x_0, we also have $b' \xrightarrow{\varepsilon : y_0} b$ in $\mathcal{B}^\varepsilon$, therefore $(b', s') \xrightarrow{\varepsilon} (b, s)$ in U.

We conclude that either $(b, s) \xrightarrow{\varepsilon} (b', s')$ or $(b', s') \xrightarrow{\varepsilon} (b, s)$, as required. $\triangleleft$

4 Union of objectives

In this section, we establish a strong form of the generalised Kopczyński conjecture for $\mathrm{BC}(\mathbf{\Sigma}_2^0)$ objectives. An objective $W \subseteq \Sigma^\omega$ is prefix-increasing[6] if for all $a \in \Sigma$ and $w \in \Sigma^\omega$, it holds that if $w \in W$ then $aw \in W$. In words, one remains in W when adding a finite prefix to a word of w. Examples of prefix-increasing objectives include prefix-independent and closed objectives.

$\blacktriangleright$ **Theorem 12.** *Let $W_1, W_2 \subseteq \Sigma^\omega$ be two $BC(\mathbf{\Sigma}_2^0)$ objectives over the same alphabet, such that W_2 is prefix-increasing. Assume that W_1 has memory $\leq k_1$ and W_2 has memory $\leq k_2$. Then $W_1 \cup W_2$ has memory $\leq k_1 k_2$.*

$\blacktriangleright$ **Remark 13.** The assumption that one of the two objectives is prefix-increasing is indeed required: for instance if $W_1 = aa(a + b)^\omega$ and $W_2 = bb(a + b)^\omega$, which are positional but not prefix-increasing, the union $(aa + bb)(a + b)^\omega$ is not positional (it has memory 2).

---

and context on history-determinism).
[6] In other papers [26, 27, 12] this notion is called prefix-decreasing, as Eve is seen as a "minimiser" player who aims to minimise some quantity.

▶ **Remark 14.** The bound $k_1 k_2$ in Theorem 12 is tight: For every k_1, k_2, there are object-ives W_1, W_2 with memories k_1, k_2 respectively, such that $W_1 \cup W_2$ has memory exactly $k_1 k_2$. One such example is as follows: let $\Sigma = \{a_1, \ldots, a_{k_1}, b_1, \ldots, b_{k_2}\}$ and $W_1 = \{w \mid w$ contains at least two different a_i infinitely often$\}$ and $W_2 = \{w \mid w$ contains at least two b_i infinitely often$\}$. We can see that W_1, W_2 and $W_1 \cup W_2$ have memory, respectively, k_1, k_2, and $k_1 \cdot k_2$ by building the Zielonka tree of these objectives and applying [19, Thms. 6, 14].

The rest of the section is devoted to the proof of Theorem 12. We explicit a construction for the union of two parity automata, inspired from the Zielonka tree of the union of two parity conditions, and show that it is $(k_1 k_2)$-wise ε-completable.

Union of parity languages.   We give an explicit construction of a deterministic parity automaton $\mathcal{T}$ recognising the union of two parity languages, which may be of independent interest. This corresponds to the automaton given by the Zielonka tree of the union (a reduced and structured version of the LAR).

We let $[d_1] = \{0, 1, \ldots, d_1\}$ and $[d_2]^* = \{1^*, \ldots, d_2^*\}$. Let $[d_1]_{\mathrm{odd}}$ and $[d_2]_{\mathrm{odd}}^*$ denote the restrictions of $[d_1]$ and $[d_2]^*$ to odd elements. By a slight abuse of notation, we sometimes treat elements in $[d_2]^*$ as natural numbers (e.g. when comparing them).

Alphabet. The input alphabet is $[d_1] \times [d_2]^*$, and we write letters as (y, z). The index of $\mathcal{T}$ is $d_1 + d_2$, we use the letter t for its output priorities.

States. States are given by interleavings of two strictly increasing sequences of $[d_1]_{\mathrm{odd}}$ and $[d_2]_{\mathrm{odd}}^*$. Any state can be taken as initial. For instance, for $d_1 = d_2 = 6$, an example of a state is:
$$\tau = \langle \underset{1}{1}, \underset{2}{3}, \underset{3}{1^*}, \underset{4}{5}, \underset{5}{3^*}, \underset{6}{5^*} \rangle.$$

We use τ to denote such a sequence, which we index from 1 to $(d_1 + d_2)/2$, and write $\tau[i]$ for its i-th element. For $y \in [d_1]$, y odd, we let $\mathsf{ind}_\tau(y) = i$ for the index such that $\tau[i] = y$. For $y \geq 2$ even, we let $\mathsf{ind}_\tau(y)$ be the index i such that $\tau[i] = y - 1$. We let $\mathsf{ind}_\tau(0) = 0$. We use the same notation for $z \in [d_2]^*$, $z \geq 1$. For example, in the state above, $\mathsf{ind}_\tau(2^*) = 3$.

The intuition is that a state stores a local order of importance between input priorities.

Transitions.   Let τ be a state and (y, z) an input letter. We define the transition $\tau \xrightarrow{(y,z):t} \tau'$ as follows:

Let $i = \min\{\mathsf{ind}_\tau(y), \mathsf{ind}_\tau(z)\}$. In the following, we assume $i = \mathsf{ind}_\tau(y)$ (the definition for $i = \mathsf{ind}_\tau(z)$ is symmetric). We let $t = 2i$, if y even, and $t = 2i - 1$ if y is odd. If y is even, we let $\tau' = \tau$. If y is odd, let i' be the smallest index $i < i'$ such that $\tau[i'] \in [d_2]^*$, and let τ' be the sequence obtained by inserting $\tau[i']$ on the left of $\tau[i]$ (or $\tau' = \tau$ if no such index i' exists). Formally,
$$\tau'[j] = \tau[j] \text{ for } j < i \text{ and } i' < j, \quad \tau'[i] = \tau[i'] \quad \text{and} \quad \tau'[j] = \tau[j-1] \text{ for } i < j \leq i'.$$

For example, for the state above, we have: $\langle 1, 3, 1^*, 5, 3^*, 5^* \rangle \xrightarrow{(3,2^*):3} \langle 1, 1^*, 3, 5, 3^*, 5^* \rangle$.

▶ **Lemma 15.** *The automaton $\mathcal{T}$ recognises the language*
$$L = \{w \in ([d_1] \times [d_2]^*)^\omega \mid \pi_1(w) \in \mathrm{Parity}_{d_1} \text{ or } \pi_2(w) \in \mathrm{Parity}_{d_2}\}.$$

Proof. We show that $L \subseteq L(\mathcal{T})$, the other inclusion is similar (and implied by Lemma 19 below). Let $(y_1, z_1)(y_2, z_2) \cdots \in W$, and assume w.l.o.g. that $y_1 y_2 \cdots \in \mathrm{Parity}_{d_1}$. Let $y_{\min} = \liminf y_i$, which is even, and let n_0 be so that for all $n \geq n_0$ we have $y_{\min} \leq y_n$. Let

$$\tau_0 \xrightarrow{(y_1, z_1):t_1} \tau_1 \xrightarrow{(y_2, z_2):t_2} \cdots$$

denote the corresponding run in $\mathcal{T}$. Let $i_n = \mathsf{ind}_{\tau_n}(y_{\min})$ be the index where $y_{\min} - 1$ appears in τ_n. Note that the sequence $(i_n)_{n \geq n_0}$ is decreasing. Let n_1 be the moment where this sequence stabilises, i.e., $i_n = i_{n_1}$ for $n \geq n_1$. By definition of the transitions of $\mathcal{T}$, for $n \geq n_1$ all output priorities are $\geq 2i_{n_1}$, and priority $2i_{n_1}$ is produced every time that a letter $(y_{\min}, z_n)$ is read. We conclude that $\mathcal{T}$ accepts w. ◄

0-**freeness of automata for prefix-increasing objectives.** The fact that W_2 is prefix-increasing will be used via the following lemma. It recasts the fact that we can add $\xrightarrow{\varepsilon:1}$ transitions everywhere to automata recognising prefix-increasing objectives.

▶ **Lemma 16.** *Let W a be prefix-increasing objective with memory $\leq k$. There exists a deterministic k-wise ε-completable automaton $\mathcal{A}$ recognising W and an ε-completion $\mathcal{A}^\varepsilon$ of $\mathcal{A}$ such that $\mathcal{A}^\varepsilon$ does not have any transition with priority 0.*

Proof. First, take any automaton $\mathcal{A}_0$ recognising W. Then let $\mathcal{A}_1$ be obtained by shifting every priority from $\mathcal{A}_0$ by 2. Clearly $\mathcal{A}_1$ also recognises W and does not have any transition with priority 0. Then apply Proposition 9 to get a k-blowup $\mathcal{A}$ of $\mathcal{A}_1$ which is k-wise ε-completable, and let $\mathcal{A}^\varepsilon$ denote the corresponding ε-completion. Since $\mathcal{A}$ has no transition with priority 0, the only possible such transitions in $\mathcal{A}^\varepsilon$ are ε-transition. Then remove all ε-transitions with priority 0 in $\mathcal{A}^\varepsilon$, and add $\xrightarrow{\varepsilon:1}$ transitions between all pairs of states in $\mathcal{A}^\varepsilon$ (in both directions).

Clearly, the obtained automaton $\tilde{\mathcal{A}}^\varepsilon$ is ε-complete and has no transition with priority 0. There remains to prove that it recognises W. Take an accepting run in $\tilde{\mathcal{A}}^\varepsilon$ and observe that the priority 1 is only seen finitely often. Hence from some moment on, the run coincides with a run in $\mathcal{A}^\varepsilon$. We conclude since W is prefix-increasing. ◄

Main proof: ε-completion of the product. We now proceed with the proof of Theorem 12. Using Theorem 7, for $l = 1, 2$, we take deterministic k_l-wise ε-completable automata $\mathcal{A}_l$ of index d_l recognising W_l, and its ε-completion $\mathcal{A}_l^\varepsilon$. For $l = 2$, we assume thanks to Lemma 16 that $\mathcal{A}_2^\varepsilon$ does not have any transition with priority 0.

We consider the product $\mathcal{A} = (\mathcal{A}_1 \times \mathcal{A}_2) \ltimes \mathcal{T}$, with states $V(\mathcal{A}_1) \times V(\mathcal{A}_2) \times V(\mathcal{T})$ and transitions $(q_1, q_2, \tau) \xrightarrow{a:t} (q_1', q_2', \tau')$ if $q_1 \xrightarrow{a:y} q_1'$ in $\mathcal{A}_1$, $q_2 \xrightarrow{a:z} q_2'$ in $\mathcal{A}_2$, and $\tau \xrightarrow{(y,z):t} \tau'$ in $\mathcal{T}$. The correctness of such a construction is folklore.[7]

▷ **Claim 17.** The automaton $\mathcal{A}$ is deterministic and recognises $W = W_1 \cup W_2$.

Therefore, there only remains to show the following lemma.

▶ **Lemma 18.** *The automaton $\mathcal{A} = (\mathcal{A}_1 \times \mathcal{A}_2) \ltimes \mathcal{T}$ is $(k_1 k_2)$-wise ε-completable.*

---

[7] $\mathcal{A}_1 \times \mathcal{A}_2$ can be seen as a Muller automaton with acceptance condition the union of two parity languages. The composition with $\mathcal{T}$ yields a correct parity automaton, as $\mathcal{T}$ recognises the acceptance condition.

The ε-completion of $\mathcal{A}$ will be a variant of a product of the form $\mathcal{A}_1^\varepsilon \times \mathcal{A}_2^\varepsilon \ltimes \overline{\mathcal{T}}$, where $\overline{\mathcal{T}}$ is a non-deterministic extension of $\mathcal{T}$ with more transitions, but which still recognises the same language.

The automaton $\overline{\mathcal{T}}$. Intuitively, we obtain $\overline{\mathcal{T}}$ by allowing to reconfigure the elements of index $> i$ in a state τ by paying an odd priority $2i - 1$, as well as allowing to move elements of $[d_1]$ to the left. We precise this idea next.

We order the states of $\mathcal{T}$ lexicographically, where we assume that $y < z$ for $y \in [d_1]$ and $z \in [d_2]^*$. Formally, we let $\tau < \tau'$ if for the first position j where τ and τ' differ, $\tau[j] \in [d_1]$ (and therefore necessarily $\tau'[j] \in [d_2]^*$). We let $\tau[..i]$ be the prefix of τ up to (and including) $\tau[i]$. We write $\tau <_i \tau'$ if $\tau[..i] < \tau'[..i]$.

Let $\tau \xrightarrow{(y,z):t}$ be a transition in $\mathcal{T}$ as above, and $i_0 = \min\{\mathsf{ind}_\tau(y), \mathsf{ind}_\tau(z)\}$ be the index determining t (i.e. $t \in \{2i_0 - 1, 2i\}$). The automaton $\overline{\mathcal{T}}$ contains a transition $\tau \xrightarrow{(y,z):t'} \tau'$ if:

1. $t' = t$ is odd, and $\tau' \leq_{i_0-1} \tau$; or
2. $t' = t$ is even, and $\tau' \leq_{i_0} \tau$; or
3. $t' \in \{2i' - 1, 2i'\}$ for some $i' \leq i_0$ and $\tau' <_{i'} \tau$. (Note that, if t is odd, this includes all (possibly even) $t' \leq t + 1$.)

In words, we are allowed to output a small (i.e. important) priority when following a strict decrease on sufficiently small components in τ. Note that transitions in $\mathcal{T}$ also belong to $\overline{\mathcal{T}}$ thanks to the rules (1) and (2).

▶ **Lemma 19.** *The automaton $\overline{\mathcal{T}}$ recognises the same language as $\mathcal{T}$.*

Proof. It is clear that $L(\mathcal{T}) \subseteq L(\overline{\mathcal{T}})$. We show the other inclusion.

Consider

$$\tau_0 \xrightarrow{(y_1,z_1):t_1} \tau_1 \xrightarrow{(y_2,z_2):t_2} \dots, \quad \text{an accepting run in } \overline{\mathcal{T}}.$$

Let $t_{\min} = \liminf t_1 t_2 \dots$, which is even. Let $i_{\min} = t_{\min}/2$ be the index responsible for producing priority $t_{\min}$. Let n_0 be such that $t_n \geq t_{\min}$ for all $n \geq n_0$. Observe that for all $n \geq n_0$, we have $\tau_n \geq_{(i_{\min}-1)} \tau_{n+1}$, and therefore there is $n_1 \geq n_0$ such that the prefix $\tau_n[..i_{\min} - 1]$ is the same for all $n \geq n_1$. In fact, the prefix $\tau_n[..i_{\min}]$ must be constant too, as we can only modify $\tau_n[i_{\min}]$ using rule (1.) and that would output priority $t_{\min} - 1$. Assume w.l.o.g. that $\tau_n[i_{\min}] = y \in [d_1]_{\mathrm{odd}}$. Now, for each $n \geq n_1$ it must be that $y_i \geq y + 1$ and it must be that $y_i = y + 1$ each time that priority $t_{\min}$ is produced. Therefore, $\liminf y_1 y_2 \cdots = y + 1$ is even. ◀

The ε-completion. We define $\mathcal{A}^\varepsilon$ as a version of the cascade product of $\mathcal{A}_1^\varepsilon \times \mathcal{A}_2^\varepsilon$ with $\mathcal{T}$, in which ε-transitions are also allowed to use the transitions of $\overline{\mathcal{T}}$. We let $\preccurlyeq$ denote the preference ordering over priorities, given by $1 \preccurlyeq 3 \preccurlyeq \dots \preccurlyeq d - 1 \preccurlyeq d \preccurlyeq \dots 2 \preccurlyeq 0$ (d even). The transitions in $\mathcal{A}^\varepsilon$ are defined as follows:

- For $a \in \Sigma$: $(q_1, q_2, \tau) \xrightarrow{a:t} (q_1', q_2', \tau')$ if this transition appears in $(\mathcal{A}_1 \times \mathcal{A}_2) \ltimes \mathcal{T}$.
- $(q_1, q_2, \tau) \xrightarrow{\varepsilon:t} (q_1', q_2', \tau')$ if $q_1 \xrightarrow{\varepsilon:y} q_1'$ in $\mathcal{A}_1^\varepsilon$, $q_2 \xrightarrow{\varepsilon:z} q_2'$ in $\mathcal{A}_2^\varepsilon$, and $\tau \xrightarrow{(y,z):t'} \tau'$ in $\overline{\mathcal{T}}$ with $t' \preccurlyeq t$.

Note that the condition $t' \preccurlyeq t$ simply allows to output a less favorable priority, so it does not create extra accepting runs. By definition, $\mathcal{A}^\varepsilon$ has been obtained by adding ε-transitions to $\mathcal{A}$. It is a folklore result that composition of non-deterministic automata also preserves the language recognised, so this construction is correct.

▷ **Claim 20.** The automaton $\mathcal{A}^\varepsilon$ recognises W.

Therefore, we there only remains to prove the following lemma.

▶ **Lemma 21.** *The automaton $\mathcal{A}^\varepsilon$ is $(k_1 k_2)$-wise ε-complete.*

The proof idea is as follows: The k_1 parts of $\mathcal{A}_1^\varepsilon$ and the k_2 parts of $\mathcal{A}_2^\varepsilon$ naturally induce a partition of the states of $\mathcal{A}^\varepsilon$ into $k_1 k_2$ parts. Then, given two states $r = (q_1, q_2, \tau)$ and $r' = (q_1', q_2', \tau')$ in the same part of $\mathcal{A}^\varepsilon$, we consider the longest common prefix of τ and τ'. We perform a case analysis: Depending on the priorities of $[d_1]$ or $[d_2]^*$ appearing in this prefix, and the transitions $q_l \xrightarrow{\varepsilon:x} q_l'$ of the automata $\mathcal{A}_1^\varepsilon$, $\mathcal{A}_2^\varepsilon$, we will find transitions $r \xrightarrow{\varepsilon:x} r'$ or $r' \xrightarrow{\varepsilon:x+1} r$ for all even x.

5   Conclusions and open questions

We characterised objectives in $\mathrm{BC}(\mathbf{\Sigma}_2^0)$ with memory (or chromatic memory) $\leq k$ as those recognised by a well-identified class of automata. In particular, this gives the first known characterisation of (chromatic) memory for ω-regular objectives, and proves that it is decidable (in fact even in **NP**). We also settled (a strengthening of) Kopczyński's conjecture for $\mathrm{BC}(\mathbf{\Sigma}_2^0)$ objectives. We now discuss some directions for future work.

Memory in finite games. This paper focuses on games over potentially infinite game graphs. A wide body of literature studies the memory over finite game graphs [24, 6]; we believe that, for ω-regular objectives, both notions should coincide, and our results should characterise the memory of ω-regular objectives over finite games too. More precisely, we believe that in item (i) in Theorem 7, it suffices to assume that W has memory $\leq k$ over games of size $f(n)$, for some finite bound $f(n)$, where n is the size of a deterministic automaton representing W.

▶ **Question 1** (Version of [32, Conjecture 9.1.2]). *Show that if an ω-regular objective has memory $\leq k$ over finite game graphs, then it has memory $\leq k$ over all game graphs.*[8]

The hypothesis on ω-regularity is necessary, as this statement already fails in the case of positionality and closed objectives [16]. As a follow-up question one could investigate the bound $f(n)$: can it be assumed polynomial?

Exact complexity of computation of memory. We established that computing the (chromatic) memory of an ω-regular objective is in **NP**. In fact, computing the chromatic memory is **NP**-hard already for simple classes of objectives, such as Muller [8] or safety ones [3]. However, no such hardness results are known for non-chromatic memory.

▶ **Question 2.** *Given a deterministic parity automaton $\mathcal{A}$ and a number k, can we decide whether the memory of $L(\mathcal{A})$ is $\leq k$ in polynomial time?*

This question is open already for the simpler case of regular open objectives (that is, those recognised by reachability automata).

---

[8] This statement is proven in the full version of this paper [13].

Assymetric 1-to-2-player lifts.   A celebrated result of Gimbert and Zielonka [22] states that if for an objective W both players can play optimally using positional strategies in finite games where all vertices belong to one player, then W is bipositonal over finite games. This result has been extended in two orthogonal directions: to objectives where both players require finite chromatic memory [6, 5] (symmetric lift for memory), and to ω-regular objectives where Eve can play positionally in 1-player games [10] (asymmetric lift for positionality). In this work, we have not provided an asymmetric lift for memory, as in most cases no such result can hold. For $\mathrm{BC}(\Sigma_2^0)$ objectives, it is known to fail already for positional objectives [21, Section 7]. For non-chromatic memory, it cannot hold for ω-regular objectives neither, because of the example described below.

▶ **Proposition 22.** *Let* $\Sigma_n = \{1, \ldots, n\}$. *For every n, the objective*

$$W_n = \{w \in \Sigma_n^\omega \mid w \text{ contains two different letters infinitely often}\}$$

has memory 2 over games where Eve controls all vertices and memory n over arbitrary games.

Proof sketch. The fact that W_n has memory n follows from [19].

   We sketch the proof that 2 memory states suffice in games G where Eve controls all vertices. For each vertex $v \in V(G)$, we let $\chi_1(v)$ and $\chi_2(v)$ be the two smallest colours in Σ_n such that there is a path from v producing them, and fix two such finite paths π_v^1, π_v^2 of minimal length. We define a 2-memory strategy for Eve as follows: when in a vertex v and memory state $j \in \{1, 2\}$, she will take the first edge from π_v^j. If this edge has colour $\chi_j(v)$, we update the memory state to $\bar{j}$. It is an easy check that this strategy ensures W_n.   ◀

In his PhD thesis, Vandenhove conjectures that an assymetric lift for chromatic memory holds for ω-regular objectives [32, Conjecture 9.1.2]. This question remains open.

▶ **Question 3.** *Is there an ω-regular objective with chromatic memory k over games where Eve controls all vertices and chromatic memory $k' > k$ over arbitrary games?*

Further decidability results for memory.   As mentioned in the introduction, many extensions of ω-automata (including deterministic ω-Turing machines and unambiguous ω-petri nets [20]) compute languages that are in $\mathrm{BC}(\Sigma_2^0)$. We believe that our characterisation may lead to decidability results regarding the memory of objectives represented by these models.

Objectives in Δ_0^3.   Some of the questions answered in this work in the case of $\mathrm{BC}(\Sigma_2^0)$ objectives are open in full generality, for instance, the generalised Kopczyński's conjecture. A reasonable next step would be to consider the class $\mathbf{\Delta}_3^0 = \mathbf{\Sigma}_3^0 \cap \mathbf{\Pi}_3^0$. Objectives in $\mathbf{\Delta}_3^0$ are those recognised by max-parity automata using infinitely many priorities [29]. Our methods seem appropriate to tackle this class, however, we have been unable to extend the extraction techniques used in the proof of Section 3.1.

─── **References** ───

   1   Udi Boker and Karoliina Lehtinen. When a little nondeterminism goes a long way: An introduction to history-determinism. *ACM SIGLOG News*, 10(1):24–51, 2023. `doi:10.1145/3584676.3584682`.

   2   Patricia Bouyer, Antonio Casares, Mickael Randour, and Pierre Vandenhove. Half-positional objectives recognized by deterministic Büchi automata. *Log. Methods Comput. Sci.*, 20(3), 2024. `doi:10.46298/LMCS-20(3:19)2024`.

3 Patricia Bouyer, Nathanaël Fijalkow, Mickael Randour, and Pierre Vandenhove. How to play optimally for regular objectives? In *ICALP*, volume 261 of *LIPIcs*, pages 118:1–118:18, 2023. `doi:10.4230/LIPIcs.ICALP.2023.118`.

4 Patricia Bouyer, Mickael Randour, and Pierre Vandenhove. The true colors of memory: A tour of chromatic-memory strategies in zero-sum games on graphs (invited talk). In *FSTTCS*, volume 250 of *LIPIcs*, pages 3:1–3:18, 2022. `doi:10.4230/LIPICS.FSTTCS.2022.3`.

5 Patricia Bouyer, Mickael Randour, and Pierre Vandenhove. Characterizing omega-regularity through finite-memory determinacy of games on infinite graphs. *TheoretiCS*, 2, 2023. `doi: 10.46298/THEORETICS.23.1`.

6 Patricia Bouyer, Stéphane Le Roux, Youssouf Oualhadj, Mickael Randour, and Pierre Vandenhove. Games where you can play optimally with arena-independent finite memory. *Log. Methods Comput. Sci.*, 18(1), 2022. `doi:10.46298/LMCS-18(1:11)2022`.

7 J. Richard Büchi and Lawrence H. Landweber. Solving sequential conditions by finite-state strategies. *Transactions of the American Mathematical Society*, 138:295–311, 1969. URL: `http://www.jstor.org/stable/1994916`.

8 Antonio Casares. On the minimisation of transition-based Rabin automata and the chromatic memory requirements of Muller conditions. In *CSL*, volume 216, pages 12:1–12:17, 2022. `doi:10.4230/LIPIcs.CSL.2022.12`.

9 Antonio Casares, Thomas Colcombet, and Karoliina Lehtinen. On the size of good-for-games Rabin automata and its link with the memory in Muller games. In *ICALP*, volume 229, pages 117:1–117:20, 2022. `doi:10.4230/LIPIcs.ICALP.2022.117`.

10 Antonio Casares and Pierre Ohlmann. Positional ω-regular languages. In *LICS*, pages 21:1–21:14. ACM, 2024. `doi:10.1145/3661814.3662087`.

11 Antonio Casares and Pierre Ohlmann. Positional ω-regular languages. *CoRR*, abs/2401.15384, 2024. `doi:10.48550/ARXIV.2401.15384`.

12 Antonio Casares and Pierre Ohlmann. Characterising memory in infinite games. *Logical Methods in Computer Science*, Volume 21, Issue 1, Mar 2025. `doi:10.46298/lmcs-21(1: 28)2025`.

13 Antonio Casares and Pierre Ohlmann. The memory of ω-regular and $BC(\Sigma_2^0)$ objectives. *CoRR*, abs/2502.05840, 2025. `doi:10.48550/ARXIV.2502.05840`.

14 Edmund M. Clarke, I. A. Draghicescu, and Robert P. Kurshan. A unified approch for showing language inclusion and equivalence between various types of omega-automata. *Inf. Process. Lett.*, 46(6):301–308, 1993. `doi:10.1016/0020-0190(93)90069-L`.

15 Edmund M. Clarke, Thomas A. Henzinger, Helmut Veith, and Roderick Bloem, editors. *Handbook of Model Checking*. Springer, Cham, 2018. `doi:10.1007/978-3-319-10575-8_2`.

16 Thomas Colcombet, Nathanaël Fijalkow, and Florian Horn. Playing safe. In *FSTTCS*, volume 29 of *LIPIcs*, pages 379–390, 2014. `doi:10.4230/LIPIcs.FSTTCS.2014.379`.

17 Thomas Colcombet, Nathanaël Fijalkow, and Florian Horn. Playing safe, ten years later. *CoRR*, abs/2212.12024, 2022. `arXiv:2212.12024, doi:10.48550/arXiv.2212.12024`.

18 Thomas Colcombet and Damian Niwiński. On the positional determinacy of edge-labeled games. *Theor. Comput. Sci.*, 352(1-3):190–196, 2006. `doi:10.1016/j.tcs.2005.10.046`.

19 Stefan Dziembowski, Marcin Jurdziński, and Igor Walukiewicz. How much memory is needed to win infinite games? In *LICS*, pages 99–110, 1997. `doi:10.1109/LICS.1997.614939`.

20 Olivier Finkel, Michał Skrzypczak, Ryszard Janicki, Slawomir Lasota, and Natalia Sidorova. On the expressive power of non-deterministic and unambiguous petri nets over infinite words. *Fundamenta Informaticae*, 183(3-4):243–291, 2022. `doi:10.3233/FI-2021-2088`.

21 Hugo Gimbert and Edon Kelmendi. Submixing and shift-invariant stochastic games, 2022. Version 2. URL: `https://arxiv.org/abs/1401.6575v2`.

22 Hugo Gimbert and Wieslaw Zielonka. Games where you can play optimally without any memory. In *CONCUR*, volume 3653, pages 428–442, 2005. `doi:10.1007/11539452_33`.

23 Yuri Gurevich and Leo Harrington. Trees, automata, and games. In *STOC*, pages 60–65, 1982. `doi:10.1145/800070.802177`.

24   Eryk Kopczyński. *Half-positional Determinacy of Infinite Games*. PhD thesis, University of Warsaw, 2008. URL: `https://www.mimuw.edu.pl/~erykk/papers/hpwc.pdf`.

25   Alexander Kozachinskiy. Energy games over totally ordered groups. In *CSL*, volume 288 of *LIPIcs*, pages 34:1–34:12, 2024. `doi:10.4230/LIPICS.CSL.2024.34`.

26   Pierre Ohlmann. *Monotonic graphs for parity and mean-payoff games. (Graphes monotones pour jeux de parité et à paiement moyen)*. PhD thesis, Université Paris Cité, France, 2021. URL: `https://tel.archives-ouvertes.fr/tel-03771185`.

27   Pierre Ohlmann. Characterizing positionality in games of infinite duration over infinite graphs. *TheoretiCS*, Volume 2, January 2023. `doi:10.46298/theoretics.23.3`.

28   Pierre Ohlmann and Michal Skrzypczak. Positionality in Σ_2^0 and a completeness result. In *STACS*, volume 289, pages 54:1–54:18, 2024. `doi:10.4230/LIPICS.STACS.2024.54`.

29   Michal Skrzypczak. Topological extension of parity automata. *Inf. Comput.*, 228:16–27, 2013. `doi:10.1016/J.IC.2013.06.004`.

30   Robert S. Streett and E. Allen Emerson. An automata theoretic decision procedure for the propositional mu-calculus. *Inf. Comput.*, 81(3):249–264, 1989. `doi:10.1016/0890-5401(89)90031-X`.

31   Wolfgang Thomas. Automata on infinite objects. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics*, 1991. URL: `https://api.semanticscholar.org/CorpusID:33060294`.

32   Pierre Vandenhove. *Strategy complexity of zero-sum games on graphs. (Complexité des stratégies des jeux sur graphes à somme nulle)*. PhD thesis, University of Mons, Belgium, 2023. URL: `https://tel.archives-ouvertes.fr/tel-04095220`.

33   Igor Walukiewicz. Pushdown processes: Games and model checking. In *CAV*, volume 1102 of *Lecture Notes in Computer Science*, pages 62–74, 1996. `doi:10.1007/3-540-61474-5_58`.