Positional Properties in Temporal Logic

Jessica Newman
University of Southampton, UK

Benjamin Plummer
University of Southampton, UK

Abstract

We study positional properties in the context of game-based reactive synthesis. Our motivation stems from having a usable specification logic, for which tractable synthesis is guaranteed. We demonstrate that every ω-regular positional property (with respect to state- or edge-labelled game graphs), is expressible in linear-time temporal logic. Additionally, we provide some necessary and sufficient conditions for when an ω-regular property is positional, and identify well-behaved subclasses of ω-regular positional properties. Using varieties of languages, we prove that no class of ω-regular positional properties can simultaneously contain a prefix-independent property and be closed under Boolean operations. We conclude by discussing the implications on alternating-time temporal logic, where we isolate a few different fragments with tractable model checking, and compare the associated expressivity of such fragments.

2012 ACM Subject Classification Theory of computation → Modal and temporal logics; Theory of computation → Automata over infinite objects; Theory of computation → Algebraic language theory

Keywords and phrases Positionality, Temporal Logic, ATL, Games on graphs

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2026.44

Related Version *Full Version with omitted proofs*: `https://arxiv.org/abs/2604.25628` [38]

Acknowledgements We thank Corina Cîrstea and Enrico Marchioni for their valuable feedback.

1 Introduction

Infinite duration, two-player graph games provide a fundamental model of the interaction between a *system* and its *environment*. The problem of *reactive synthesis* (generating a system controller which implements a specification) can be phrased as computing a winning strategy in such a game. This technique quickly becomes intractable, and is highly sensitive to the (logical) formalism chosen to express the desired property of the system. Even when using the property-specification logic linear-time temporal logic (LTL) [43], which has powerful automata-theoretic tools, the synthesis problem is famously 2EXPTIME-complete [9]. This provides a major obstacle to applying synthesis in practical applications.

The difficulty of synthesis is inextricably linked to the vast number of possible strategies: on any non-trivial game graph (even with a single player), there are an infinite number of possible strategies, because strategies can use an arbitrary amount of memory. If one could impose a maximum bound on the amount of memory that a strategy can use, we could reduce the space of strategies to be finite (at least for games with a finite state space). This, in turn, reduces LTL synthesis to LTL *model checking* - which is PSPACE-complete [47] - by guessing a strategy which uses memory less than the bound, and then performing model checking on each induced transition system. Notice that this discussion also carries over to model checking of the *alternating-time temporal logic* ATL* [2] interpreted over game structures, whereby combining the classical labelling algorithm (see [1]) with LTL model checking the path formula (which performs LTL synthesis in the manner just described) enables us to also

reduce ATL* model checking to LTL model checking. However, in both cases (LTL synthesis and ATL* model checking) this approach is doomed to fail, as the amount of memory that winning strategies may require is unbounded.

In this work, we investigate to what extent we can syntactically restrict the collection of LTL formulas (or, more generally, the path formula in ATL*), to obtain a class of properties whose winning strategies' memory can be bounded by a single state - i.e. properties where *positional* strategies suffice. LTL is recognised as having a good balance between *expressiveness* and *usability*, and our key result (Theorem 26) is a positive one: *every ω-regular positional property is expressible in* LTL. As we shall see, which properties are positional is sensitive to the kind of games which we are doing synthesis over, specifically to whether labels (sometimes called colours) appear on the *edges* or on the *states*. Our result is with respect to the larger class of positional properties: those which are positional over *state-labelled* games. These are a natural class of properties in ATL*, because atomic propositions are assigned to states.

Recently, (edge-labelled) positional ω-regular objectives have received a full characterisation in [18], providing different classes of automata which recognise them. One key observation in loc. cit. is that a positional language L always has a *totally ordered set of residuals* (recall that, for some finite word u, the residual is the set containing all suffixes v such that uv is in L). This requirement lends itself to an *automata-theoretic* perspective, as characterisations can be built on top of the automaton of residuals, which has as states the residuals of the language. However, the requirement of totally ordered residuals can instead be presented as a purely syntactic closure condition on the language[1]. The authors of [18] note that a *language-theoretic* characterisation, using conditions of this form, is left open. We complement their results by providing such a language-theoretic characterisation of positional ω-regular languages over edge-labelled arenas in terms of additional closure conditions on the language (Corollary 10). This has two main benefits: firstly, these conditions can be phrased algebraically, enabling the use of tools from algebraic language theory (we pursue this in Section 5). Secondly, they naturally extend to the setting of state-labelled positionality (Theorem 24), which is less clear in the automata-theoretic case. We further investigate this in Proposition 25, finding that *a property is positional over state-labelled arenas when it can be played optimally with memory of the previous label in edge-labelled arenas.*

As mentioned above, we find that LTL can express all ω-regular positional properties. However, this does not tell us much about how to concretely specify such properties. Therefore, in the latter half of the paper we use the language-theoretic characterisation to continue the investigation of specification languages for positional properties. Firstly, we obtain a precise characterisation of positional ω-regular *prefix-independent* languages (Proposition 13), and provide a concrete form for specifying a wide range of such languages. We then investigate whether we can isolate a *language variety* (of infinite words) consisting entirely of positional properties. This would give us a good candidate for a logical specification language to express positional properties, as language varieties are closed under Boolean operations. Unfortunately, we present a negative result in Corollary 30, that: *any (non-trivial) variety of languages consisting only of positional properties does not contain any prefix-independent properties.* This immediately rules out the existence of formalisms for expressing positional properties that both support combining positional properties via logical connectives and include even basic positional prefix-independent properties such as Büchi objectives. Our final endeavour is to present several possible restrictions of the path formula in ATL*, which ensure the resulting fragments can only express positional properties. We isolate

[1] if a word $uv \in L$ and a word $xy \in L$, then either $xv \in L$ or $uy \in L$.

these fragments using the previously obtained characterisation, discuss the associated model checking problems, and give a fine-grained analysis of their expressiveness, comparing them to the logics EATL and EATL$^+$ [24], and providing examples of which positional properties can/cannot be expressed in each fragment.

Related Work. Determining which objectives in infinite games are positional has been well-studied, with a prominent example being that parity objectives are bipositional [25].

A particularly interesting question is how to characterise which objectives are positional within a given class. Several sufficient conditions have been found for positionality in the class of prefix-independent objectives, including in [30] and later [7]. However, a full characterisation of prefix-independent positional objectives remains open. It has been found, however, that the bipositional prefix-independent objectives are all parity objectives [21].

In terms of positional ω-regular objectives, some sufficient conditions were given in [31], and a more general characterisation of memory requirements in [12]. Further results on memory requirements for different objectives can be found in [48]. As mentioned above, some full characterisations of positional ω-regular languages have been presented in [18].

Further classes where positionality has been characterised include languages recognised by deterministic Büchi automata [11], safety games [19], and the Borel class Σ_2^0 [40]. A general characterisation of positionality in terms of universal graphs was given in [39].

A full characterisation of bipositional objectives (for finite games) was provided in [27].

The positionality of model checking has been previously noted for fragments of temporal logic such as ATL [14], and there has been some analysis on restricting certain variants of ATL* to positional strategies [46, 15]. It has also been found that all formulae in ATL$^+$, a fragment with non-positional path formulae, can be expressed in ATL, a positional fragment [28]. In a similar spirit, some fragments of Strategy Logic have been identified in which the strategies required are less complex than the full Strategy Logic [37, 26].

In a similar spirit to the motivation of our work on temporal logic, GR(1) synthesis [10] syntactically restricts LTL specifications to obtain a polynomial-time synthesis algorithm (we comment on the relationship between our approaches in Remark 35).

Contributions & Organisation. We start, in Section 2, by fixing definitions and notation, and recalling basic facts we use about: two-player games, positional properties, language theory, and temporal logic. In Section 3, we investigate *edge-labelled* positional properties, we discuss: expressing positional properties in LTL (Section 3.1); a language-theoretic characterisation of ω-regular positionality (Section 3.2); and how this characterisation simplifies in the case of prefix-independent ω-regular languages (Section 3.3). Section 4 then relates our findings to *state-labelled* positionality: we give a language-theoretic characterisation for this more general class of properties (Section 4.1), and show how our expressibility result transfers to the state-labelled setting (Section 4.2). In Section 5 we prove our no-go theorem stating that no ∞-variety can exist which consists of positional properties and contains a prefix-independent property. In Section 6, we describe how one can perform efficient model checking for positional fragments of ATL*, and propose two such fragments. Finally, Section 6.2 gives various inexpressibility results, establishing a hierarchy including several variants of ATL.

We now outline our contributions, we:

- Prove that all ω-regular languages which are positional over edge-labelled, and state-labelled, games are expressible in LTL (Theorems 5 and 26).

- Obtain a language-theoretic characterisation for when an ω-regular language is positional over edge-labelled, and state-labelled, games (Corollary 10 and Theorem 24). We note the former has also been independently proven by Colcombet and Idir [20]. Furthermore, we show this can be simplified when the language is prefix-independent (Proposition 13).
- Provide a polynomial-time decision procedure to decide whether a Wilke algebra recognises a positional language (Corollary 11).
- Give a concrete description of a large class of positional prefix-independent languages (Proposition 16).
- Show that there is no language variety consisting entirely of positional languages which contains a prefix-independent language (Corollary 30).
- Establish that the ATL^* model checking problem (which can state LTL synthesis) for positional fragments is in PSPACE, and for bipositional fragments is in Σ_2^p (Proposition 32).
- Present several positional fragments of ATL^* and establish an expressivity hierarchy among them, and other variants of ATL (Theorem 37).

Most proofs have been omitted due to space limitations; some key proofs can be found in Appendix A and the remainder in the full version [38].

2 Preliminaries

Words. Fix an alphabet Σ. We use Σ^*, Σ^+, Σ^ω, and Σ^∞ to denote the set finite words, finite non-empty words, infinite words, and possibly infinite words (i.e. $\Sigma^\infty := \Sigma^\omega \cup \Sigma^*$), respectively. Note we can take a relation as the alphabet, and then words will be sequences of pairs in the relation. We write $w[i..j]$ for sub-word of w from index i to index j inclusive, for any word $w \in \Sigma^\infty$ which has length greater than or equal to i and j. We write $w[i] := w[i, i]$.

ω-regular languages. ω-regular languages can be equivalently defined, inter alia, as the languages recognised by finite ω-semigroups and finite Büchi automata [17], expressible in the monadic second-order logic S1S [45], and given by ω-regular expressions [36]. They include languages expressible in LTL [49]. Two ω-regular languages are equivalent exactly when they have the same set of ultimately periodic words, i.e. words of the form uv^ω for $u, v \in \Sigma^*$ [16]. Given a language $L \subseteq \Sigma^\omega$ and word $u \in \Sigma^*$, we can obtain the residual $R_L(u) = \{v \in \Sigma^\omega \mid uv \in L\}$. For ω-regular languages, the set $R(L) = \{R_L(u) \mid u \in \Sigma^*\}$ of residuals is finite [34]. We say a language L has totally ordered residuals when $R(L)$ is totally ordered by inclusion. A language L is *prefix-independent* when for any word in the language, we can add or remove any finite prefix and remain in the language, i.e. for all $x \in \Sigma^\omega$, for any $u \in \Sigma^*$, $x \in L$ iff $ux \in L$.

Arenas. Two-player games are played on *arenas*: tuples (V_1, V_2, δ) where V_1 and V_2 are finite sets of states controlled by Player 1 and Player 2 respectively, and $\delta \subseteq V \times V$ is a total relation, where we write V for the disjoint union of V_1 and V_2. Given an edge $e \in \delta$ where $e = (v, v')$, we define $src(e) := v$ and $tgt(e) := v'$. Our arenas will implicitly come equipped with a *labelling*, where labels comes from a set Σ. These come in two forms: *edge-labelled* and *state-labelled*. A *state-labelled arena* comes equipped with a function $\pi : V \to \Sigma$, assigning labels to states. An *edge-labelled arena* replaces the above δ with a total relation $\delta \subseteq V \times (\Sigma \times V)$, associating each edge with a label, note there can now be multiple edges between states (but with different labels). Given an element $e \in \delta$ such that $e = (v, c, v')$, we define $src(e) := v$, $tgt(e) := v'$ and $col(e) := c$. When drawing diagrams of arenas, we will sometimes draw an edge $x \to y$ labelled with a word $a_0 \dots a_n \in \Sigma^+$, when it determines a unique sequence of transitions $x \xrightarrow{a_0} \cdots \xrightarrow{a_n} y$ in the arena.

Games & Strategies. Given a (labelled) arena (V_1, V_2, δ), a *strategy* is a function $\sigma : V \cup \delta^+ \to \delta$ such that $src(\sigma(v)) = v$ and $\sigma(v) \in \delta$ for all $v \in V$, and $src(\sigma(we)) = tgt(e)$ and $\sigma(we) \in \delta$ for all $w \in \delta^*$ and $e \in \delta$. A strategy selects for each state an edge which is accessible from that state, and for each nonempty sequence of edges an edge which is accessible from the target state of the final edge in that sequence. We define a *positional strategy* as a function $\sigma : V \to \delta$, such that $src(\sigma(v)) = v$ and $\sigma(v) \in \delta$ for all $v \in V$. Note that our definitions of strategy are a little nonstandard, as they pick moves at states controlled by both players. The players are differentiated in the following definition of *play*.

Given an arena (V_1, V_2, δ), an initial state $q \in V$, and two strategies σ_1, σ_2, a *play* $\varsigma \in \delta^\omega$ is defined inductively below.

$$\varsigma[0] = \begin{cases} \sigma_1(q) & \text{if } q \in V_1 \\ \sigma_2(q) & \text{if } q \in V_2 \end{cases} \qquad \varsigma[i+1] = \begin{cases} \sigma_1(\varsigma[0,i]) & \text{if } tgt(\varsigma[i]) \in V_1 \\ \sigma_2(\varsigma[0,i]) & \text{if } tgt(\varsigma[i]) \in V_2 \end{cases}$$

The definition is analogous for positional strategies, but we only require the previous state $tgt(\varsigma[i])$ rather than the play so far. Every play ς generates an associated *trace* $\lambda \in \Sigma^\omega$. For state-labelled arenas, we can set $\lambda[i] = \pi(src(\varsigma[i]))$. For edge-labelled arenas, we can set $\lambda[i] = col(\varsigma[i]))$. We will denote the play generated by strategies σ_1, σ_2 as $play(\sigma_1, \sigma_2, q)$ and the trace as $trace(\sigma_1, \sigma_2, q)$. Given a labelled arena, a strategy σ, and an initial state q, we define the *Player 1 outcome of* σ as the set $out_1(\sigma, q) := \{trace(\sigma, \sigma', q) \mid \sigma' \text{ is a strategy}\}$. The *Player 2 outcome of* σ $out_2(\sigma, q)$ can be defined analogously (this time the first component in *trace* will be quantified over).

A *property* is a subset of traces $L \subseteq \Sigma^\omega$. We can assign a property as the *winning condition*, or *objective*, of a game played over an arena. Given an arena with winning condition L and initial state q, we say Player 1 *is (positionally) winning* iff there is a (positional) strategy σ_1 such that $out_1(\sigma_1, q) \subseteq L$. Player 2 is *(positionally) winning* iff they have a (positional) strategy σ_2 such that $out_2(\sigma_2, q) \subseteq \Sigma^\omega \setminus L$. It is well known that ω-regular languages are *determined* winning conditions [13]: Player 1 is winning iff Player 2 is not winning.

Positionality. A property L is *state-labelled (edge-labelled) positional*, if whenever Player 1 has a strategy σ in some state-labelled (edge-labelled) arena with objective L where σ is winning for P1 in all states $w \in W$ for some $W \subseteq V$, there exists a positional stategy σ' such that P1 winning from all $w \in W$ also. A property L is *state-labelled (edge-labelled) bipositional* if both L and $\Sigma^\omega \setminus L$ are state-labelled (edge-labelled) positional. We can restrict the notions of positionality to arenas where all states are controlled by Player 1 (where $V_2 = \emptyset$). The resulting notions are *1P edge-labelled positional* and *1P state-labelled positional* properties. For positionality restricted to 1P arenas, we differentiate between *uniform* and *non-uniform* positionality: uniform positionality is as defined above, whereas a property is *non-uniform state-labelled (edge-labelled) positional* if whenever Player 1 is winning at a state q in some state-labelled (edge-labelled) arena with objective L, they are also positionally winning from q. It can be seen that uniform positionality implies non-uniform positionality, and additionally for positionality over 2P arenas this distinction makes no difference, so either definition can be taken. When some of these terms are omitted and not clear from context, we take the default to be *uniform* positionality, and to be over 2P arenas.

Memory. Strategies $\sigma : V \cup \delta^+ \to V$ as defined above can be implemented with *memory structures*. A memory structure is a deterministic automaton $\mathcal{M} = (M, m_0, \mu)$ where M is a set of memory states, $m_0 \in M$ is an initial state, and $\mu : M \times \Sigma$ is an update function, paired with a strategy $\sigma : V \times M \to \delta$ with the same constraints on chosen edges as the

strategies above. When generating a play, the move chosen by the strategy is $\sigma(v, m)$ for the current state v and current memory state m. Every time a label $c \in \Sigma$ is seen, the current memory state is updated from m to $\mu(m, c)$. In other words, given a partial play $\varsigma[0, i]$ ending on a state in V_1, and a corresponding trace $\lambda[0, i]$, $\varsigma[i+1] = \sigma(tgt(\varsigma[i]), \mu^+(m_0, \lambda[0, i]))$ where μ^+ is the extension of μ to finite sequences of labels (i.e. for $w \in \Sigma^+$ $\mu^+(m, w) = \mu(\dots\mu(\mu(m, w[0]), w[1])\dots w[|w|-1])$). Winning a game with an ω-regular objective requires only a finite-memory strategy [13]. Positional strategies are those implemented by memory structures where $|M| = 1$.

Algebraic Language Theory. We recall standard notions from algebraic language theory (full details can be found in e.g. [41]). An *ω-semigroup* is a pair $S = (S_+, S_\omega)$ of a semigroup S_+ and a set S_ω, equipped with a *mixed product* $S_+ \times S_\omega \to S_\omega$ and *infinite product* $\pi : S_+^\omega \to S_\omega$. We say S *recognises* a language $L \subseteq \Sigma^\infty$ if there is a homomorphism $\varphi : \Sigma^\infty \to S$ from the free ω-semigroup $\Sigma^\infty = (\Sigma^*, \Sigma^\omega)$ to S such that for some $F \subseteq S_+ \cup S_\omega$, $\varphi^{-1}(F) = L$. Finite ω-semigroups recognise exactly the (ω-)regular languages, and admit finite presentations as *Wilke algebras*, in which the infinite product is determined by a map $(-)^\omega : S_+ \to S_\omega$.

A *variety* of ω-semigroups is a class of ω-semigroups closed under subsemigroups, quotients, and finite products. An *∞-variety* of languages is a class of recognisable languages closed under Boolean operations, residuals, and preimages of morphisms between free ω-semigroups. Varieties of ω-semigroups are in bijective correspondence with ∞-varieties of languages [42].

Linear-time Temporal Logic. LTL is a property-specification logic, used to specify subsets of infinite words. We define the syntax of LTL over our set of labels Σ, by taking the smallest collection of formulas such that each $a \in \Sigma$ is a formula, and $\neg\varphi$, $\varphi \vee \psi$, $X\varphi$, and $\varphi U \psi$ are all formulas when φ and ψ are formulas. The semantics of LTL are subsets of infinite words Σ^ω, defined in the standard way (see, e.g. [22, Chapter II, 6]). A key fact we shall use in Section 3, is that LTL (when the semantics is taken over possibly infinite words) expresses exactly the *star-free languages* [23]: those recognisable by finite aperiodic monoids. We shall use *ω-star-free* to mean the part of the class which consists of languages of infinite words.

It can be useful to consider the semantics of LTL over 1P state-labelled arenas, where we take Σ to be 2^{Prop}, the set of subsets of atomic propositions (these are known as *serial Kripke frames* to modal logicians). Here there are two choices, the *existential* or *universal* semantics: depending on whether we are looking for a single path through the transition system which generates the formula as a trace, or we demand that every possible trace through the transition system satisfies the formula.

3 Edge-Labelled Positional Omega-Regular Languages

We refer to edge-labelled positional properties as positional properties in this section.

3.1 Expressibility in LTL

We will prove the statement that all ω-regular positional properties are ω-star-free (or equivalently, expressible as LTL formulae). Our approach is to characterise the ω-star-free languages as those which are *counter-free* and *ω-counter-free* (we define these notions shortly). The benefit of this, is that being (ω-)counter-free is a language-theoretic property, which gives us direct methods to show that they follow from a language being positional (in Lemma 3 and Lemma 4).

▶ **Definition 1.** *A language $L \in \Sigma^\omega$ is*
- *(CF) counter-free whenever $uv^n w \in L$ iff $uv^{n+m}w \in L$ for all $m \in \mathbb{N}$,*
- *(ω-CF) ω-counter-free whenever $u(vx^n y)^\omega \in L$ iff $u(vx^{n+m}y)^\omega \in L$ for all $m \in \mathbb{N}$,*

for some $n \in \mathbb{N}$ and all $u, v, x, y \in \Sigma^*$ and $w \in \Sigma^\omega$.

A language being counter-free enforces that it behaves aperiodically on the finite prefixes of infinite words, whereas it being ω-counter-free enforces that it behaves aperiodically on the factors of ultimately periodic words. It is well know that ω-star-free languages are those which can be recognised by finite aperiodic monoids [23, Theorem 1.1], which we can show enforce the same aperiodicity conditions. Note that the only if direction of Proposition 2 relies on the Arnold congruence from [4].

▶ **Proposition 2.** *An ω-regular language $L \subseteq \Sigma^\omega$ is ω-star-free iff it is counter-free and ω-counter-free.*

Recall that the set of residuals for a language is: totally ordered when the language is positional [18, Lemma 4.1]; and finite when the language is ω-regular. These two properties are enough to show that a language is counter-free. Roughly, if residuals are totally ordered by inclusion, then the residuals of any word uv^i form a monotone chain, so membership of a word $uv^i w$ in L can change at most once; otherwise two residuals would be incomparable. But a non-counter-free language forces words where this change happens at arbitrarily high i, yielding arbitrarily many distinct residuals, contradicting finiteness.

▶ **Lemma 3.** *If a language $L \subseteq \Sigma^\omega$ has a finite set of residuals $R(L)$ which is totally ordered by inclusion, then it is counter-free.*

We have a similar result for ω-counter-freeness, by first showing that for each word $u(vw^i x)$, there is a bound on i at which the word no longer changes membership of L, and then showing this bound is uniform using the syntactic congruence from [4].

▶ **Lemma 4.** *If a language $L \subseteq \Sigma^\omega$ is ω-regular and positional, then it is ω-counter-free.*

Combining Proposition 2, Lemma 3, and Lemma 4, we obtain:

▶ **Theorem 5.** *If a language $L \subseteq \Sigma^\omega$ is ω-regular and positional, then it is ω-star-free.*

Since LTL can express all ω-star-free languages (and vice versa), then all positional properties can be expressed in LTL.

3.2 Language-Theoretic Characterisation

In this section we will provide a characterisation for ω-regular properties which are positional over edge-labelled arenas. Unless otherwise stated, in this section we will consider arenas to be finite-state. We shall use the fact that for a game with an ω-regular winning condition, a player has a winning strategy iff they have a winning strategy with memory that can be encoded in finite states [13]. In a 1P arena, this will generate an ultimately periodic play, i.e. a play of the form pl^ω for some $p \in \delta^*, l \in \delta^+$. The main idea is to enforce closure conditions on L which guarantee that for any winning ultimately periodic play pl^ω, there exists a winning ultimately periodic play $p'l'^\omega$ where $p'l'$ contains strictly fewer violations of positionality - choices of different edges from same state - than pl; this is done with closure conditions that guarantee for any cycle in the play beginning and ending on the same state, either we can remove this cycle or take this cycle forever and still have a winning play. Since pl is finite, it contains a finite number of violations of positionality, so we end up with an

ultimately periodic play in which any given state is always followed by the same edge. From this we can construct a positional strategy which generates the same play. We can find some closure conditions which achieve this and are also necessary for positionality. This method lets us characterise *non-uniform* 1P positional properties:

▶ **Proposition 6.** *An ω-regular language L is non-uniform 1P positional*[2] *iff for all $u, y \in \Sigma^*$, $v, w, x \in \Sigma^+$:*
(1) *if $uvwx^\omega \in L$ then either $uv^\omega \in L$ or $uwx^\omega \in L$*
(2) *if $u(vwy)^\omega \in L$ then either $uvw^\omega \in L$ or $u(vy)^\omega \in L$*

The closure under union of certain classes of positional language has been a notable question since [32]. It can be seen that when we take a finite union of such languages, the resulting language will still satisfy the closure conditions (1) and (2). Therefore:

▶ **Proposition 7.** *ω-regular non-uniform 1P positional languages are closed under finite union.*

However, there are languages in this class which are not positional on two-player arenas. For example, the property $L = \{a^\omega, b^\omega\}$, which says that either we only see a or we only see b; the residuals $R_L(a) = \{a^\omega\}$ and $R_L(b) = \{b^\omega\}$ are incomparable. For games without a given starting state, we require that the language has totally ordered residuals (see Section 3.1). A finite-memory strategy now generates an ultimately periodic play from each state - this means we can have violations of positionality "across" the plays. In conjunction with the method from Proposition 6, these can be amended using totally ordered residuals, which given two partial plays ending at the same state allows us to replace the continuation of one play with the other.

▶ **Proposition 8.** *An ω-regular language L is uniform 1P positional iff it has totally ordered residuals and satisfies (1) and (2) from Proposition 6.*

In [18], it was shown that uniform 1P positional ω-regular languages are exactly the positional ω-regular languages.

▶ **Proposition 9** ([18])**.** *An ω-regular language is positional over finite Player 1 controlled edge-labelled arenas iff it is positional over all edge-labelled arenas.*

So from this we have our main result for this section, that the same conditions hold for positionality over 2P arenas.

▶ **Corollary 10.** *An ω-regular language L is positional over 2P edge-labelled arenas (finite or infinite) iff it has totally ordered residuals and satisfies (1) and (2) from Proposition 6.*

We note that a very similar form of the above characterisation has been independently discovered by Colcombet and Idir [20].

We will discuss some corollaries of this characterisation. Firstly, these conditions allow us to give a decision procedure for checking whether an ω-regular language is positional. Given a finite algebraic object recognising a ω-regular language, we can easily check the above conditions from the multiplication table. Therefore:

[2] Note the results from the previous section do not necessarily apply to non-uniform 1P positional languages, as this class of languages does not require totally ordered residuals.

▶ **Corollary 11.** *Given a Wilke algebra $S = (S_+, S_\omega)$ recognising a language $L \subseteq \Sigma^\omega$, we can determine whether L is positional in polynomial time.*

Unfortunately, unlike in the non-uniform 1P case, languages positional over 2P arenas are not closed under finite union in general, as the union of two languages with totally ordered residuals may not necessarily have totally ordered residuals. However, we can take a finite union in restricted circumstances:

▶ **Corollary 12.** *Given a finite set $\mathbf{L}$ of positional ω-regular languages, the union $\cup\mathbf{L}$ is positional iff $\cup\mathbf{L}$ has totally ordered set of residuals.*

Prefix-independent ω-regular languages are such that $R(L) = \{L\}$ [3]. Therefore, if we have an arbitrary $\omega-$regular language L_1 and a prefix-independent ω-regular language L_2, then $R(L_1 \cup L_2) = \{X \cup L_2 | X \in R(L_1)\}$. Therefore, if $R(L_1)$ is totally ordered then so is $R(L_1 \cup L_2)$, so a special case of the above corollary is the (known [18]) fact that the union of a positional ω-regular language and prefix-independent ω-regular language will be positional.

3.3 Prefix-Independent Languages

The class of prefix-independent languages, where only the infinite behaviour of a word affects membership, is important in verification. We provide a condition characterising the positional ω-regular prefix-independent languages, and follow with a concrete description of a class of such languages. Applying our characterisation in Corollary 10, we find that prefix-independent languages trivially meet many of the conditions:

▶ **Proposition 13.** *A prefix-independent ω-regular language L is positional iff $(uv)^\omega \in L$ implies either $u^\omega \in L$ or $v^\omega \in L$.*

Note that this provides an alternate justification of the known result [18, Theorem 3.4] that the class of prefix-independent ω-regular positional languages is closed under finite union. The condition in Proposition 13 is similar in form to *concavity*, a known sufficient condition for positionality of prefix-independent languages [32]:

▶ **Definition 14** (Concavity). *For all $x, y \in \Sigma^\omega$ such that $x = x_1x_2\ldots$, $y = y_1y_2\ldots$ and for all i, $x_i, y_i \in \Sigma^+$, if $x_1y_1x_2y_2\ldots \in L$ then either $x \in L$ or $y \in L$.*

However, concavity is not necessary for positionality, even for ω-regular prefix-independent languages, as can be seen by the following example:

▶ **Example 15.** $L = \{w \in \Sigma^\omega |$ substring bb occurs finitely often in $w\}$ is prefix-independent, positional, but not concave, as $(ab)^\omega \in L$ but neither $(aabb)^\omega$ nor $(bbaa)^\omega$ are.

So, we can see the condition in Proposition 13 as a weakening of concavity to be both necessary and sufficient for positionality. We will use this characterisation to capture a large class of positional prefix-independent ω-regular languages. An ω-regular language is prefix-independent iff it can be expressed in the form $\Sigma^*(R_1)^\omega + \ldots + \Sigma^*(R_n)^\omega$ [3]. Since prefix-independent positional languages are closed under finite union [18], we will focus on languages of the form $\Sigma^*(R)^\omega$. The following form can express many interesting properties:

▶ **Proposition 16.** *Suppose a language L can be expressed in the form $\Sigma^*(SR)^\omega$, where $S \subseteq \Sigma$, and $R \subseteq (\Sigma \setminus S)^*$ is a subword-closed regular language with totally-ordered residuals. Then, L is prefix-independent and positional.*

This encodes a condition with both "liveness" and "safety", in the sense that S gives a set of labels which are seen infinitely often and R restricts the behaviour between these labels. We will provide some examples of languages in this class. In fact, Example 15 above can be expressed in this form, where $S = \Sigma \setminus \{b\}$ and $R = \{b, \varepsilon\}$. Let us explain what is meant by R being *subword-closed* [6]. The subword ordering on a language L is defined such that $u \leq_{SUB} v$ iff there is an increasing sequence $k_1, k_2, \dots k_{|u|}$ such that $v[k_1]v[k_2] \dots v[k_{|u|}] = u$ - i.e. all the characters of u can be found within v in the correct order. We can specify a subword-closed language by giving the set of minimal disallowed subwords - a word is in the language iff it does not contain any of these as a subword. The set of disallowed subwords is called the *anti-dictionary*. The requirement for positionality places some restrictions on the entries in the anti-dictionary:

▶ **Proposition 17.** *Suppose $L \subseteq \Sigma^*$ is a subword-closed language with anti-dictionary D. L has totally-ordered residuals iff for every $xy, uv \in D$, where $x, y, u, v \in \Sigma^*$, there is some $d \in D$ such that $d \leq_{SUB} xv$ or $d \leq_{SUB} uy$.*

With this we can define languages where we disallow a single specific subword:

▶ **Proposition 18.** *Every subword-closed language with a single element anti-dictionary has totally-ordered residuals.*

And we can define languages where we disallow a set of labels:

▶ **Proposition 19.** *If D is the anti-dictionary of a subword-closed language with totally-ordered residuals, then $D \cup \{a\}$ for any $a \in \Sigma$ is the anti-dictionary of a subword-closed language with totally-ordered residuals.*

A Rabin condition is a finite collection of pairs (U_i, V_i), where U_i must be visited infinitely often, and V_i must be visited finitely often. Since we can encode each V_i as single-character entries in an anti-dictionary, we can represent these conditions as finite unions of languages of the form Proposition 16 - this also subsumes parity conditions. Some further examples of interest of subword-closed languages with totally ordered residuals are found in [29].

4 State-Labelled Positional Omega-Regular Languages

4.1 Language-Theoretic Characterisation

We can characterise state-labelled ω-regular positional properties with similar conditions to those that characterise edge-labelled positional properties. Recall that a strategy is not positional whenever it visits the same state twice and prescribes different moves each time. In state-labelled arenas, since each state has a single label, we know that if we visit the same state twice the label seen will be the same each time. So, our closure conditions only need to apply when we see the same label twice, as otherwise we have not seen the same state twice.

▶ **Proposition 20.** *An ω-regular language L is non-uniform 1P state-labelled positional iff the following conditions hold for all $u, y \in \Sigma^*$, $v, w, x \in \Sigma^+$ where $v[0] = w[0]$:*
(1) *if $uvwx^\omega \in L$ then either $uv^\omega \in L$ or $uwx^\omega \in L$*
(2) *if $u(vwy)^\omega \in L$ then either $uvw^\omega \in L$ or $u(wy)^\omega \in L$*

This can be shown with the same method as Proposition 6. These weaker requirements for positionality allow for more properties to be positional. For example, we can have properties that contain permutations of distinct labels, since violations of positionality can only occur when we see the same label twice.

▶ **Example 21.** The language $(abc)^\omega$ is 1P state-labelled positional.

In fact, the language in this example is 1P uniform positional. Despite this, the language does not have a totally ordered set of residuals. This requirement is also weaker for state-positional languages; we only require that for words ending in the same label, the set of residuals is totally ordered.

▶ **Proposition 22.** *If L is a state-labelled positional ω-regular language, then for each $a \in \Sigma$, the set of residuals for words ending in a, $\{R_L(ua)|u \in \Sigma^*\}$, is totally ordered by inclusion.*

We can now make the analogous statement of Proposition 8, which can be proven with the same method.

▶ **Proposition 23.** *L is uniform 1P state-labelled positional iff for each $a \in \Sigma$, the set $\{R_L(wa)|w \in \Sigma^*\}$ is totally ordered by inclusion, and conditions (1) and (2) hold from Proposition 20.*

Additionally, as seen previously in the edge-labelled case, the conditions for 1P uniform positionality also characterise positionality on 2P arenas (often called a 1-to-2 player lift), although demonstrating this is quite involved: our plays now form trees, so we have to show these conditions allow us to replace trees of winning plays with trees of winning plays containing fewer violations of positionality. Interestingly, the requirement for totally ordered residuals does most of the "work" here, allowing us to, given two subtrees of plays rooted at the same state, replace one with the other. The other conditions enforce a similar property to progress consistency in [18]; once we have shown we can recursively nest a subtree within itself an arbitrary number of times and still have a winning play, these allow us to extend this nesting to an infinite depth and still have a winning play.

▶ **Theorem 24.** *L is positional over finite 2P state-labelled arenas iff for each $a \in \Sigma$, the set $\{R_L(wa)|w \in \Sigma^*\}$ is totally ordered by inclusion, and conditions (1) and (2) hold from Proposition 20.*

4.2 Relation to Edge-Labelled Positionality

There is a connection between edge-labelled positional and state-labelled positional objectives. It is known that edge-labelled positional languages can be played positionally on state-labelled arenas, and that the converse is not true - there are more state-labelled positional objectives than edge-labelled [32]. We make precise the converse relation between them:

▶ **Proposition 25.** *An ω-regular language L is positional over state-labelled arenas iff it can be played optimally with memory of the previous label in edge-labelled arenas.*

Here, "memory of the previous label" means a strategy $\sigma : V \times \Sigma \to V$, where when generating a play, the current state s and label of the previous edge c is given to σ to obtain the successor state $\sigma(s, c)$. This can be seen in the extension of the conditions for edge-labelled positionality (Corollary 10) to state-labelled positionality (Theorem 24) - since we can "see" the previous label in state-labelled arenas, we only need that these conditions hold at points where the previous labels were equal. Proposition 13 can be extended to state-labelled positionality in the same way, giving us that the class of prefix-independent state-labelled positional languages is closed under finite union.

Similarly to the edge-labelled case, state-labelled positional ω-regular languages are expressible in LTL. This can be shown through a very similar argument as in Section 3.1. Note that the proof of Lemma 3 also holds for languages where just the sets $\{R_L(ua)|u \in \Sigma^*\}$

are totally ordered by inclusion for each $a \in \Sigma$, since uv^i will end in the same character as uv^j for any $i, j > 0$. For the analogue to Lemma 4, it is simple to adapt the given counterexample to a state-labelled game, from which the same argument follows. Therefore:

▶ **Theorem 26.** *If a language $L \subseteq \Sigma^\omega$ is ω-regular and positional over state-labelled arenas, then it is ω-star-free.*

5 Varieties

In the following section we will use "positional" to refer to properties positional over edge-labelled arenas. LTL allows us to compositionally build formulae to specify properties, using temporal operators and Boolean operations. However, these operations do not respect positionality; for example: $Fp \wedge Fq$ is not positional, even though both Fp and Fq are. We might wonder whether it is possible to construct some similar specification language closed under Boolean operations which is restricted to expressing only positional languages, thereby providing a compositional syntax for specifying positional properties. To this end, we will investigate varieties of positional languages - classes of positional languages closed under certain combinations of Boolean operations. It is known that varieties of languages correspond to varieties of ω-semigroups, so the (non)existence of varieties of ω-semigroups that recognise only positional languages will determine the (non)existence of Boolean-closed classes of positional languages. An overview of recognition of languages by ω-semigroups can be found in [41], and an overview of the correspondence between language varieties and varieties can be found in [42]. For algebraic convenience, we will be deal with sets of mixed finite and infinite words, i.e. subsets of Σ^∞. However, our claims will only focus on the infinite part of the languages, so for any definition relating to ω-languages (prefix-independence, positionality, etc.), we take $L \subseteq \Sigma^\infty$ to meet these definitions when $L \cap \Sigma^\omega$ meets these definitions.

Firstly, we find that prefix-independent languages are closed under Boolean operations, taking residuals, and pre-images of free ω-semigroup morphisms, so form an ∞-variety. Furthermore, each corresponding ω-semigroup (S_+, S_ω) has the property that the mixed product mapping $S_+ \times S_\omega \to S_\omega$ is right projection, so $uv \mapsto v$ for all $u \in S_+, v \in S_\omega$. This gives us the following:

▶ **Proposition 27.** *Prefix-independent languages form an ∞-variety corresponding to the identity $uv = v$ for $u \in \Sigma^*, v \in \Sigma^\omega$.*

However, if we attempt to find within this a variety consisting of *positional* prefix-independent languages, we can only find trivial examples:

▶ **Proposition 28.** *If V is an ∞-variety consisting entirely of prefix-independent positional languages, then the only ω-languages it contains are $\emptyset$ and Σ^ω.*

▶ Remark 29. Since closure under complement means these languages are bipositional, and bipositional prefix-independent languages encode parity conditions [21], this can equally be phrased as the nonexistence of a non-trivial class of parity conditions closed under Boolean operations.

This places some restrictions on varieties of positional languages in general. Suppose we have a variety $\mathcal{V}$ entirely consisting of positional languages. If this contained a prefix-independent language L, the variety generated by L must be prefix-independent (by Proposition 27) and must be a subvariety of $\mathcal{V}$. This would give us a variety of prefix-independent languages, which we know to be impossible by the above. Therefore:

▶ **Corollary 30.** *An ∞-variety containing only positional languages must not contain a (non-trivial) prefix-independent language.*

However if we weaken our closure requirements, we can find a positive ∞-variety (an ∞-variety minus the requirement to be closed under complement) of prefix-independent positional languages:

▶ **Proposition 31.** *The class of languages containing, for each alphabet Σ, the languages of the form $FG(A_1) \cup \ldots \cup FG(A_n)$ where $A_i \subseteq \Sigma$ form a positive ∞-variety.*

This suggests we may be able to find more specification languages for positional properties if we drop the need for closure under all Boolean operations.

6 Applications to Temporal Logic

In this section we apply the above characterisation of positional properties to identify fragments of the temporal logic ATL^* in which we can assume agents use positional strategies. We first introduce some notions specific to this section.

ATL^* extends LTL by allowing *path formulae*, defined in LTL, to be bound to modalities $\langle C \rangle$ which determine (in the semantics) a set of subsets of traces from a state, according to what the coalition of agents C can force. A formula $\langle C \rangle \varphi$, where φ is an LTL formula, will be satisfied at a state x when *there exists* a subset of traces that C can force from x, such that *every* trace within satisfies φ. An implicit parameter of this logic is the set of agents Ag.

In this paper we often consider fragments of ATL^* generated by a fragment of LTL which are allowed as the path formulae. To this end, we define the syntax of the logic as parameterised over a subset $\mathcal{A}$ of LTL formula which possibly contain free variables. The grammar for $\mathsf{ATL} + \mathcal{A}$ is separated into *state formula* ψ and path formula φ. For any $p \in Prop, C \subseteq Ag, \alpha(x_1, \ldots, x_n) \in \mathcal{A}$:

$$\psi ::= p \mid \psi \vee \psi \mid \neg\psi \mid \langle C \rangle \varphi \qquad\qquad \varphi ::= X\psi \mid \psi U \psi \mid \psi R \psi \mid \alpha[\psi_1 \ldots \psi_n]$$

The semantics of the logic is taken over *concurrent game structures*, we refer to [22, Chapter II, 9] for details. When $\mathcal{A} = \emptyset$ we get plain ATL; when $\mathcal{A}$ consists of all LTL formulae possibly with free variables we obtain the full logic ATL^*; when we take $\mathcal{A}$ to contain FG and GF (with free variables in the obvious places) we get EATL; when we take $\mathcal{A}$ to contain Boolean combinations of X, U, and R we get ATL^+; and when we take $\mathcal{A}$ to take Boolean combinations of X, U, R, GF, and FG we get EATL^+. When we impose that there is precisely one agent, we get CTL, CTL^*, ECTL, CTL^+, and ECTL^+, using the values of $\mathcal{A}$ in the previous sentence. In Section 6 we will consider fragments $\mathsf{ATL} + \mathcal{A}$ where we take $\mathcal{A}$ to contain single formula with free variables, we shall abuse notation and omit the set notation and the free variables; for example, we will say $\mathsf{ATL} + GU$ to mean ATL with the path formula $G(\psi U \psi)$ added to the syntax.

We can give a *positional semantics* for $\mathsf{ATL} + \mathcal{A}$ by taking the subsets of traces from a state x as the subsets which arise from *positional strategies* for C. Furthermore, we can consider a *bipositional semantics*, when each trace in each subset (which arises from a positional C-strategy) arises from a positional strategy for $\mathsf{Ag} \setminus C$. It is a well-known fact that the positional semantics of ATL agrees with the regular semantics which use memoryful strategies (in fact, the bipositional semantics of ATL also agrees with the regular semantics). We call some instance of $\mathsf{ATL} + \mathcal{A}$ a *(bi)positional fragment* if across all expressible formulae and models the regular semantics agrees with the (bi)positional semantics.

6.1 Positional Fragments

In the domain of temporal logic, Corollary 30 tell us that we must restrict the path formula in ATL* drastically to obtain a *positional semantics* - one where only positional strategies are considered - which agrees with the memoryful semantics. Despite this, we examine two such restrictions. We consider two fragments of ATL* where we add $GF\psi_1 \wedge FG\psi_2$, and $G(\psi_1 U \psi_2)$, to the path formula of ATL respectively, as described above. We see these as natural choices of path formula, because in one-agent ATL (i.e. CTL), these give us the ability to express Rabin objectives (which subsume parity, and maintain positionality wrt the first player), and Rabin objectives *with* a safety objective, respectively. These claims are substantiated shortly in Proposition 36.

As mentioned in the introduction, we achieve efficient model checking for restrictions of ATL* with positional path formula, by reducing the problem to LTL model checking (which is in PSPACE). Given a formula φ in the positional fragment of LTL, to determine whether φ holds at a state x, it suffices to (non-deterministically) enumerate positional Player 1 strategies in the game, and do *universal* model checking on the LTL formula φ on the resulting transition system at x. We can say more when the formula φ is bipositional. First note that *existential* LTL model checking of a positional path formulae φ is in NP, as we can guess a positional strategy to generate an ultimately periodic trace pl^ω where $|pl|$ is at most the number of states, which can be checked in time $O(|pl| \cdot |\varphi|)$[35]. So, if φ is bipositional, then $\neg\varphi$ is positional and checking $A\varphi \equiv \neg E\neg\varphi$ is in co-NP. Therefore, ATL* model-checking of bipositional properties is in Σ_2^p (i.e. in NP given a co-NP oracle [5]), by guessing a positional strategy for P1 and using the oracle to verify $A\varphi$. These facts yield the following:

▶ **Proposition 32.** *The model checking problem for any positional fragment of* ATL* *is in PSPACE. The model checking problem for any bipositional fragment of* ATL* *is in* Σ_2^p.

We now verify that adding various path formula of interest to ATL maintains positionality of the semantics.

▶ **Proposition 33.** *If we add any of the following composite* LTL *modalities to the path formula of* ATL*, the semantics remains positional:* $GF\psi_1 \wedge FG\psi_2$, $G(\psi_1 U \psi_2)$, $GF\psi_1 \wedge FG\psi_2 \wedge G\psi_3$, *and* $G(\psi \vee X\psi)$.

Proof sketch. The first formula generates a prefix-independent set of traces, so it suffices to verify the conditions of Proposition 13. For the remaining three, the claim follows from verifying the conditions in Corollary 10, as every property positional on edge-labelled arenas is positional on state-labelled arenas. ◀

When adding in two of the path formula considered in Proposition 33 to ATL, it turns out we obtain logics of equal expressive power.

▶ **Proposition 34.** ATL + $GF \wedge FG \wedge G$ *and* ATL + GU *are mutually expressive.*

Proof sketch. The following equivalences can be routinely verified: $\langle C\rangle(GF\psi \wedge FG\varphi \wedge G\phi) \equiv \langle C\rangle(\phi U(\langle C\rangle(G((\varphi \wedge \phi)U(\psi \wedge \varphi \wedge \phi)))$ and $\langle C\rangle(G\psi U\varphi) \equiv \langle C\rangle(GF\varphi \wedge FG\top \wedge G(\psi \vee \varphi))$. ◀

▶ Remark 35. GR(1) synthesis [10] restricts LTL specifications to an implication from a conjunction of Büchi conditions to a conjunction of Büchi conditions, to obtain a polynomial-time synthesis algorithm. Our positional fragments are incomparable to this approach, as $\mathsf{FG}p \wedge \mathsf{GF}q$ is not directly expressible in GR(1), and we cannot express conjunctions of Büchi conditions as they are not positional (and are inexpressible in our fragments, see 42). We leave a detailed comparison to future work.

Over models with a single player, we now show that adding $GF\varphi \wedge FG\psi$ ($G\varphi U\psi$) to path formula lets us express Rabin conditions (mixed Rabin and safety conditions). Recall a Rabin condition is a finite collection of pairs (U_i, V_i), where U_i must be visited infinitely often, and V_i must be visited finitely often. We say we can *encode* a Rabin condition in a fragment of CTL* when, given U_i and V_i are denoted by propositional variables u_i and v_i, we can write a formula which holds precisely when the Rabin condition is satisfied. In a similar fashion, we can speak of safety conditions, where some set W must never be visited, being encoded in a fragment of CTL*. The following proposition relies on disjunctions distributing over E in CTL*, thus may not apply to the corresponding fragments of ATL*.

▶ **Proposition 36.** *The fragment* $\mathsf{CTL} + GF \wedge FG$ *can encode a Rabin condition. The fragment* $\mathsf{CTL} + GU$ *can encode a Rabin condition combined with a safety condition.*

Proof. We show the second claim. Given a Rabin condition (u_i, v_i) indexed by a finite set I and a safety condition w (both encoded as atomic propositions), we can write a formula equivalent to $E(\bigvee_{i\in I}(GFu_i \wedge FGv_i) \wedge Gw)$, using distributivity of E over $\vee$, and Proposition 34. ◀

Note that parity conditions are subsumed by Rabin conditions, so can be expressed by both fragments in Proposition 36.

6.2 Inexpressivity results

In the remainder of this section, we make various inexpressivity claims for our different positional fragments, along with EATL (where GF and FG path formula are permitted), and EATL^+ (where we further allow Boolean combinations of path formula). We collect the claims in the following theorem, the proofs of which will follow.

▶ **Theorem 37.** *Let* $>$ *denote the strictly more expressive than relation, we have the following:*

$$\mathsf{ATL} \overset{[28]}{=} \mathsf{ATL}^+ \overset{(39)}{<} \mathsf{EATL} \overset{(40)}{<} \mathsf{ATL} + GF \wedge FG \overset{(41)}{<} \mathsf{ATL} + GU \overset{(42)}{<} \mathsf{EATL}^+ \overset{(43)}{<} \mathsf{ATL}^*$$

Furthermore, EATL^+ *cannot express all state-labelled positional properties.*

Note that EATL is positional whereas EATL^+ is not, so as far as we are aware $\mathsf{ATL}{+}GU$ is the largest known fragment of ATL* containing only positional path formulae.

Recall that the standard procedure for showing inexpressibility results in temporal logic (see e.g. [22, Lemma 10.3.2]) is to show that a formula in one logic picks out a class of models, which cannot be expressed by the other. It turns out, that to compare our fragments, it is often sufficient to perform this procedure on just the one-agent restrictions of our logics:

▶ **Lemma 38.** *Let* A *be a set of (possibly composite)* LTL *modalities, and* φ *be an* LTL *formula. Suppose we have two families of rooted transition systems* $\{\mathcal{M}_n\}_{n\in\mathbb{N}}$ *and* $\{\mathcal{N}_n\}_{n\in\mathbb{N}}$*, for which* $\mathcal{M}_n \vDash \exists\varphi$ *and* $\mathcal{N}_n \nvDash \exists\varphi$ *for all* n*, and where* $\mathcal{M}_n$ *and* $\mathcal{N}_n$ *cannot be distinguished by any* $CTL + A$ *formula of modal depth less than* n*. This is enough to show that* $\langle i\rangle\varphi$ *is not expressible in* $\mathsf{ATL} + \mathcal{A}$*, for some agent* $i \in \mathsf{Ag}$*.*

In [24, Theorem 4.4], the authors show that $CTL^+ < ECTL$ by proving $E(GFp)$ is not expressible in CTL^+ using the assumption of Lemma 38, thus we have as a corollary:

▶ **Corollary 39.** $\langle i\rangle GFp$ *is not expressible in* ATL^+*.*

To separate EATL and $\mathsf{ATL} + GF \wedge FG$, we can show the assumptions of Lemma 38 are satisfied by adapting the proof of [33, Lemma 2.8.4], where it is shown that $E(GFp \wedge Gq)$ distinguishes two classes of models for which no ECTL formula can. It turns out that $E(GFp \wedge FGq)$ also distinguishes the same two classes of models.

▶ **Proposition 40.** *$\langle i \rangle(GFp \wedge FGq)$ is not expressible in* EATL.

It easily follows from Proposition 34, that $\mathsf{ATL} + GU$ is as at least as expressive as $\mathsf{ATL} + GF \wedge FG$. To show it is strictly more, we can construct two classes of rooted transitions systems which satisfy Lemma 38 by hand.

▶ **Proposition 41.** *$\langle i \rangle(GpUq)$ is not expressible in $\mathsf{ATL} + GF \wedge FG$.*

We can easily see EATL^+ is at least as expressive as $\mathsf{ATL} + GU$, by using Proposition 34 and the fact that $GF\varphi \wedge FG\psi \wedge G\psi$ is a Boolean combination of EATL path formulas. To get that EATL^+ is strictly more expressive, we can show the models given in the proof of $ECTL < ECTL^+$ in [24, Theorem 4.3] cannot be distinguished by $\mathsf{ATL} + GU$ formulas, and apply Lemma 38.

▶ **Proposition 42.** *$\langle i \rangle(GFp \wedge GFq)$ is not expressible in $\mathsf{ATL} + GU$.*

The formula $EG(p \vee Xp)$ was shown not to be expressible in $ECTL^+$ in [24, Theorem 4.2]. Again, we can use two classes of models to apply Lemma 38, thus we obtain:

▶ **Corollary 43.** *$\langle i \rangle G(p \vee Xp)$ is not expressible in EATL^+*

This corollary tells us two things: firstly that ATL^* is strictly more expressive than EATL^+, and secondly that not all positional properties are expressible in EATL^+ (we saw in Proposition 33 that $G(p \vee Xp)$ is a positional property).

7 Conclusion

We have shown that positional ω-regular languages are ω-star-free, and provided a language-theoretic characterisation of these languages to complement the automata-theoretic characterisation of [18]. We have applied these results to problems in temporal logic. Firstly, a negative result showing we cannot find a property specification language closed under Boolean operations which expresses only positional and prefix-independent properties. Secondly, we have presented various fragments of ATL^* with positional path formulae (and so more tractable model-checking), and shown where they fall expressively in the hierarchy of ATL^* fragments.

We present directions for future work. Further work is required to refine exactly where positional ω-regular languages are situated within the various hierarchies of sub-ω-regular languages. We also do not know whether there are ω-regular 1P non-uniform positional languages which are not ω-star-free. On another note, totally-ordered residuals are a key requirement for positionality, but they are not entirely understood. Some work has been done for the finite case in [29], where they are termed "Ferrers languages". Due to Corollary 12, it would be good to know exactly when the union of two languages has totally ordered residuals. It would also be interesting to know whether there are any non-trivial varieties consisting entirely of positional languages. Additionally, in the domain of finite languages, there are variety theorems for several restricted combinations of Boolean operations, e.g. disjunctive varieties [44], basic varieties [8]. If these can be extended to ω-languages, then we can find classes of positional languages which are instances of these weaker types of variety, which may provide canonical algebraic identities to classify these languages. Finally, since ATL^+ is expressible in ATL [28], there may be an analogue of Proposition 36 which holds for ATL^* through a similar construction.

References

1 E. Allen Emerson and Chin-Laung Lei. Modalities for model checking: branching time logic strikes back. *Science of Computer Programming*, 8(3):275–306, 1987. doi:10.1016/0167-6423(87)90036-0.

2 Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. Alternating-time temporal logic. *J. ACM*, 49(5):672–713, September 2002. doi:10.1145/585265.585270.

3 Dana Angluin and Dana Fisman. Regular ω-languages with an informative right congruence. *Information and Computation*, 278:104598, 2021. Special Issue on Ninth International Symposium on Games, Automata, Logics and Formal Verification (GandALF 2018). doi:10.1016/j.ic.2020.104598.

4 André Arnold. A syntactic congruence for rational ω-languages. *Theoretical Computer Science*, 39:333–335, 1985. Third Conference on Foundations of Software Technology and Theoretical Computer Science. doi:10.1016/0304-3975(85)90148-3.

5 Sanjeev Arora and Boaz Barak. *The polynomial hierarchy and alternations*, pages 95–105. Cambridge University Press, 2009.

6 A. Atminas and V. Lozin. Deciding atomicity of subword-closed languages. *Theoretical Computer Science*, 1003:114595, 2024. doi:10.1016/j.tcs.2024.114595.

7 Alessandro Bianco, Marco Faella, Fabio Mogavero, and Aniello Murano. Exploring the boundary of half-positionality. *Annals of Mathematics and Artificial Intelligence*, 62(1):55–77, June 2011. doi:10.1007/s10472-011-9250-1.

8 Fabian Birkmann, Stefan Milius, and Henning Urbat. On language varieties without boolean operations. In *Language and Automata Theory and Applications: 15th International Conference, LATA 2021, Milan, Italy, March 1–5, 2021, Proceedings*, pages 3–15, Berlin, Heidelberg, 2021. Springer-Verlag. doi:10.1007/978-3-030-68195-1_1.

9 Roderick Bloem, Krishnendu Chatterjee, and Barbara Jobstmann. Graph games and reactive synthesis. In Edmund M. Clarke, Thomas A. Henzinger, Helmut Veith, and Roderick Bloem, editors, *Handbook of Model Checking*, pages 921–962. Springer International Publishing, Cham, 2018. doi:10.1007/978-3-319-10575-8_27.

10 Roderick Bloem, Barbara Jobstmann, Nir Piterman, Amir Pnueli, and Yaniv Sa'ar. Synthesis of reactive(1) designs. *Journal of Computer and System Sciences*, 78(3):911–938, 2012. In Commemoration of Amir Pnueli. doi:10.1016/j.jcss.2011.08.007.

11 Patricia Bouyer, Antonio Casares, Mickael Randour, and Pierre Vandenhove. Half-Positional Objectives Recognized by Deterministic Büchi Automata. In Bartek Klin, Sławomir Lasota, and Anca Muscholl, editors, *33rd International Conference on Concurrency Theory (CONCUR 2022)*, volume 243 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 20:1–20:18, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CONCUR.2022.20.

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

13 J. Richard Büchi and Lawrence H. Landweber. Solving sequential conditions by finite-state strategies. *Transactions of the American Mathematical Society*, 138(0):295–311, 1969. doi:10.1090/s0002-9947-1969-0280205-0.

14 Nils Bulling and Wojciech Jamroga. Comparing variants of strategic ability: how uncertainty and memory influence general properties of games. *Autonomous Agents and Multi-Agent Systems*, 28(3):474–518, May 2014. doi:10.1007/s10458-013-9231-3.

15 Simon Busard, Charles Pecheur, Hongyang Qu, and Franco Raimondi. Reasoning about memoryless strategies under partial observability and unconditional fairness constraints. *Information and Computation*, 242:128–156, 2015. doi:10.1016/j.ic.2015.03.014.

16 Hugues Calbrix, Maurice Nivat, and Andreas Podelski. Ultimately periodic words of rational ω-languages. In Stephen Brookes, Michael Main, Austin Melton, Michael Mislove, and David Schmidt, editors, *Mathematical Foundations of Programming Semantics*, pages 554–566, Berlin, Heidelberg, 1994. Springer Berlin Heidelberg.

17 Olivier Carton, Dominique Perrin, and Jean-Eric Pin. Automata and semigroups recognizing infinite words. In E. Grädel J. Flum and T. Wilke, editors, *Logic and Automata, History and perspectives*, pages 585–596. Amsterdam University Press, 2007. URL: https://hal.science/hal-00340797.

18 Antonio Casares and Pierre Ohlmann. Positional ω-regular languages. *TheoretiCS*, Volume 5, February 2026. doi:10.46298/theoretics.26.5.

19 Thomas Colcombet, Nathanaël Fijalkow, and Florian Horn. Playing safe, ten years later. *Logical Methods in Computer Science*, Volume 20, Issue 1, January 2024. doi:10.46298/lmcs-20(1:10)2024.

20 Thomas Colcombet and Olivier Idir. An algebraic characterisation of eve-positional languages, 2026. doi:10.48550/arXiv.2604.22648.

21 Thomas Colcombet and Damian Niwiński. On the positional determinacy of edge-labeled games. *Theoretical Computer Science*, 352(1):190–196, 2006. doi:10.1016/j.tcs.2005.10.046.

22 Stéphane Demri, Valentin Goranko, and Martin Lange. *Temporal Logics in Computer Science: Finite-State Systems*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 2016. doi:10.1017/CBO9781139236119.

23 Volker Diekert and Paul Gastin. First-order definable languages. In Jörg Flum, Erich Grädel, and Thomas Wilke, editors, *Logic and Automata: History and Perspectives [in Honor of Wolfgang Thomas]*, Texts in Logic and Games, pages 261–306. Amsterdam University Press, 2008.

24 E. Allen Emerson and Joseph Y. Halpern. "Sometimes" and "not never" revisited: on branching versus linear time (preliminary report). In *Proceedings of the 10th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages*, POPL '83, pages 127–140, New York, NY, USA, 1983. Association for Computing Machinery. doi:10.1145/567067.567081.

25 E.A. Emerson and C.S. Jutla. Tree automata, mu-calculus and determinacy. In *[1991] Proceedings 32nd Annual Symposium of Foundations of Computer Science*, pages 368–377, 1991. doi:10.1109/SFCS.1991.185392.

26 Patrick Gardy, Patricia Bouyer, and Nicolas Markey. Dependences in Strategy Logic. In Rolf Niedermeier and Brigitte Vallée, editors, *35th Symposium on Theoretical Aspects of Computer Science (STACS 2018)*, volume 96 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 34:1–34:15, Dagstuhl, Germany, 2018. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.STACS.2018.34.

27 Hugo Gimbert and Wiesław Zielonka. Games where you can play optimally without any memory. In Martín Abadi and Luca de Alfaro, editors, *CONCUR 2005 – Concurrency Theory*, pages 428–442, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

28 Aidan Harding, Mark Ryan, and Pierre-Yves Schobbens. Approximating atl* in atl. In Agostino Cortesi, editor, *Verification, Model Checking, and Abstract Interpretation*, pages 289–301, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg. doi:10.1007/3-540-47813-2_20.

29 Mustapha Kabil and Maurice Pouzet. Injective envelopes of transition systems and ferrers languages. *RAIRO-Theor. Inf. Appl.*, 54:4, 2020. doi:10.1051/ita/2020005.

30 Eryk Kopczyński. Half-positional determinacy of infinite games. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Automata, Languages and Programming*, pages 336–347, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

31 Eryk Kopczyński. Omega-regular half-positional winning conditions. In Jacques Duparc and Thomas A. Henzinger, editors, *Computer Science Logic*, pages 41–53, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

32 Eryk Kopczyński. *Half-positional determinacy of infinite games*. PhD thesis, University of Warsaw, 2014.

33 François Laroussinie. *Logique temporelle avec passé pour la spécification et la vérification des systèmes réactifs*. PhD thesis, Grenoble INP, 1994. Thèse de doctorat dirigée par Schnoebelen, Philippe Informatique Grenoble INPG 1994. URL: http://www.theses.fr/1994INPG0136.

34 Oded Maler and Ludwig Staiger. On syntactic congruences for ω-languages. *Theoretical Computer Science*, 183(1):93–112, 1997. Formal Language Theory. doi:10.1016/S0304-3975(96)00312-X.

35 Nicolas Markey and Philippe Schnoebelen. Model Checking a Path (Preliminary Report). In Amadio, Roberto M., Lugiez, and Denis, editors, *Lecture Notes in Computer Science*, volume 2761 of *Lecture Notes in Computer Science*, pages 251–265, Marseilles, France, 2003. Springer. doi:10.1007/b11938.

36 Robert McNaughton. Testing and generating infinite sequences by a finite automaton. *Information and Control*, 9(5):521–530, 1966. doi:10.1016/S0019-9958(66)80013-X.

37 Fabio Mogavero, Aniello Murano, and Luigi Sauro. A behavioral hierarchy of strategy logic. In Nils Bulling, Leendert van der Torre, Serena Villata, Wojtek Jamroga, and Wamberto Vasconcelos, editors, *Computational Logic in Multi-Agent Systems*, pages 148–165, Cham, 2014. Springer International Publishing. doi:10.1007/978-3-319-09764-0_10.

38 Jessica Newman and Benjamin Plummer. Positional properties in temporal logic, 2026. doi:10.48550/arXiv.2604.25628.

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

40 Pierre Ohlmann and Michał Skrzypczak. Positionality in Σ^0_2 and a Completeness Result. In Olaf Beyersdorff, Mamadou Moustapha Kanté, Orna Kupferman, and Daniel Lokshtanov, editors, *41st International Symposium on Theoretical Aspects of Computer Science (STACS 2024)*, volume 289 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 54:1–54:18, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.STACS.2024.54.

41 Dominique Perrin and Jean Éric Pin. Ii - automata and semigroups. In Dominique Perrin and Jean Éric Pin, editors, *Infinite Words*, volume 141 of *Pure and Applied Mathematics*, pages 75–131. Elsevier, 2004. doi:10.1016/S0079-8169(04)80003-5.

42 Dominique Perrin and Jean Éric Pin. Vi - varieties. In Dominique Perrin and Jean Éric Pin, editors, *Infinite Words*, volume 141 of *Pure and Applied Mathematics*, pages 265–306. Elsevier, 2004. doi:10.1016/S0079-8169(04)80007-2.

43 Amir Pnueli. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science (sfcs 1977)*, pages 46–57, 1977. doi:10.1109/SFCS.1977.32.

44 Libor Polák. A classification of rational languages by semilattice-ordered monoids. *Archivum Mathematicum*, 040(4):395–406, 2004. URL: http://eudml.org/doc/249314.

45 J. Richard Büchi. Symposium on decision problems: On a decision method in restricted second order arithmetic. In Ernest Nagel, Patrick Suppes, and Alfred Tarski, editors, *Logic, Methodology and Philosophy of Science*, volume 44 of *Studies in Logic and the Foundations of Mathematics*, pages 1–11. Elsevier, 1966. doi:10.1016/S0049-237X(09)70564-6.

46 Pierre-Yves Schobbens. Alternating-time logic with imperfect recall. *Electronic Notes in Theoretical Computer Science*, 85(2):82–93, 2004. LCMAS 2003, Logic and Communication in Multi-Agent Systems. doi:10.1016/S1571-0661(05)82604-0.

47 A. P. Sistla and E. M. Clarke. The complexity of propositional linear temporal logics. *J. ACM*, 32(3):733–749, July 1985. doi:10.1145/3828.3837.

48 Pierre Vandenhove. *Strategy complexity of zero-sum games on graphs*. Theses, Université Paris-Saclay ; Université de Mons (Belgique), April 2023. URL: https://theses.hal.science/tel-04095220.

49 Pierre Wolper. Temporal logic can be more expressive. *Information and Control*, 56(1):72–99, 1983. doi:10.1016/S0019-9958(83)80051-5.

A Omitted Proofs

▶ **Lemma 3.** *If a language $L \subseteq \Sigma^\omega$ has a finite set of residuals $R(L)$ which is totally ordered by inclusion, then it is counter-free.*

Proof. Suppose we have a language $L \subseteq \Sigma^\omega$ which has a finite, totally ordered set of residuals $R(L)$. We will first show that for any choice of $u, v \in \Sigma^*$, $w \in \Sigma^\omega$ the word uv^iw can only "switch" membership of L at most once as i increases from 0. In other words, if $uv^0w = uw \in L$ and there is some $uv^iw \notin L$, then $uv^jw \in L$ for $j < i$ and $uv^kw \notin L$ for $k \geq i$, and vice versa. To show this, first suppose this was not the case. We begin with the case where $uw \in L$. We would have some $i, j \in \mathbb{N}$ with $0 < i < j$ such that $uv^iw \notin L$ and $uv^jw \in L$. If we take i and j to be the least such numbers where the language switches membership, we know that for $0 \leq p < i$, $uv^pw \in L$, and for $i \leq q < j$, $uv^qw \notin L$. Let us look at the residuals for the words $uv^{(i-1)}$ and $uv^{(j-1)}$. We know that $w \in R_L(uv^{(i-1)})$ and $w \notin R_L(uv^{(j-1)})$, as $uv^{(i-1)}w \in L$ but $uv^{(j-1)}w \notin L$. So, $R_L(uv^{(i-1)}) \not\subseteq R_L(uv^{(j-1)})$. We know that $vw \notin R_L(uv^{(i-1)})$ and $vw \in R_L(uv^{(j-1)})$, as $uv^iw \notin L$ but $uv^jw \in L$. Therefore, $R_L(uv^{(j-1)}) \not\subseteq R_L(uv^{(i-1)})$. So, the set of residuals cannot be totally ordered, contradicting our assumption. The argument for the case where $uw \notin L$ is analogous. Therefore, it must be that for any choice of u, v, w, the word uv^iw switches membership at most once as i increases from 0.

Given this, we will assume that L is not counter-free. Therefore, for all n, there exists some $u, v, w, m \in \mathbb{N}$ such that $uv^nw \in L$ iff $uv^{n+m}w \notin L$. So for any n, we can find a word with substring v where the switch in membership happens at some point after v is iterated n times. Let us call the point at which the membership switches m_n for each n. Then we have the following, where $m_n > n$:

$n = 0, u_0w_0 \in L, u_0v_0^{m_0}w_0 \notin L$

$n = 1, u_1v_1w_1 \notin L, u_1v_1^{m_1}w_1 \in L$

$n = 2, u_2v_2^2w_2 \notin L, u_2v_2^{m_2}w_2 \in L$

...

Although this example is just illustrative; the switch in membership may happen in the other direction for any n.

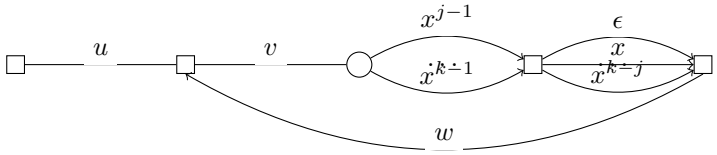
Note that for a given n, we generate at least n different residuals. To see this, suppose for some $n = i$, $uv^iw \in L$ and $uv^{m_i}w \notin L$. We know membership of the language can only switch once, so $uv^jw \in L$ for $j < m_i$ and $uv^jw \notin L$ for $j \geq m_i$. Then we can look at the residuals for each uv^k where k ranges from $0 \leq k \leq m_i$. We can see that $uv^{k+l}w \in L$ iff $k + l < m_i$. This means the residual $R_L(uv^k)$ must be distinct from any residual $R_L(uv^{k-j})$ for $1 \leq j \leq k$, as the latter contains $v^{m_i-k}w$ whereas the former does not, as this would take it out of the language. So, each of the residuals $R_L(uv^0)$ up to $R_L(uv^k)$ are distinct sets. We can argue symmetrically for the case when $uv^iw \notin L$ and $uv^{m_i}w \in L$.

Since the set of residuals $R(L)$ must at least contain each of these residuals for each n, it must be infinite. If it were finite, say of size k, then it could not contain all of the distinct residuals for the case of $n = k + 1$. This contradicts our assumption that $R(L)$ is finite, so L must in fact be counter-free. ◀

▶ **Lemma 4.** *If a language $L \subseteq \Sigma^\omega$ is ω-regular and positional, then it is ω-counter-free.*

Proof. Take a language $L \subseteq \Sigma^\omega$ which is ω-regular and positional. Take some arbitrary $u, v, w, x \in \Sigma^*$. We will first show that a word $u(vx^nw)^\omega$ cannot be out of L at some point $n = i$, in the language at some point $n = j$ where $j > i$, and again out of the language at some point $n = k$ where $k > j$. This means that for every word $u(vx^nw)^\omega$, there is a finite n at which increasing n no longer changes membership of the language. First, for some $u, v, w, x \in \Sigma^*$ take i to be some value where $u(vx^iw)^\omega \notin L$, take j to be the first value

such that $j > i$ and $u(vx^jw)^\omega \in L$, and take k to be the first value such that $k > j$ and $u(vx^kw)^\omega \notin L$. Let us construct the following 2-player game with L as a winning condition, where P1 controls the square nodes:



Note that P1 has a memoryful strategy which guarantees all traces are in L, by always aiming for a total of x^{k-1}. However, no choice of positional strategy can guarantee that P1 wins, as P2 can always force either x^{j-1}. or x^k However, this is contradictory as we know by assumption that L is positional - so it must be the case there is always some value n after which $u(vx^nw)^\omega$ does not change membership of L.

Since L is ω-regular, we can use the syntactic congruence $\approx$ of [4] to obtain a finite monoid that recognises L. Suppose this monoid is of size k. For each element $[x] \in M$, it must be that there is $[x]^n = [x]^m$ for some $n \leq k$, $n \leq m \leq k$. Furthermore, $[x]^n = [x]^{n+i(m-n)}$ for all i. So, iterating $[x]$ eventually gives us a cycle of period $m - n$. Since $[x]^n = [x]^m = [x^n] = [x^m]$, this also means that $x^n \approx x^{n+i(m-n)}$, so for any $u, v, w \in \Sigma^*$, $u(vx^nw)^\omega \in L$ iff $u(vx^{n+i(m-n)}w)^\omega \in L$.

We will show that for any word of the form $u(vx^iw)^\omega$, the i after which it no longer switches from being in L to outside of L or vice versa is bounded by $k = |\Sigma^*/\approx| + 1$. Let n, m be the points at which $[x]^n = [x]^m$ as detailed above. Suppose the i at which the word switches occurs on $i = (n + j(m - n))$ for some j. Since, by the above, $u(vx^nw)^\omega \in L$ iff $u(vx^{n+i(m-n)}w)^\omega \in L$, the switch would have to happen at $u(vx^nw)^\omega$, and $n < k$. Suppose the i at which the final switch occurs at some $n + j(m - n) < i < n + (j + 1)(m - n)$ for some $j > 0$. Either $u(vw^{(n+j(m-n))}x)^\omega$ (and therefore also $u(vw^{(n+(j+1)(m-n))}x)^\omega$) is in L and $u(vw^ix)^\omega$ is not in L or vice versa. However, as i is the value of the final switch, $u(vw^{(n+(j+1)(m-n))}x)^\omega$ must be in L iff $u(vw^ix)^\omega$ is in L, which contradicts the previous statement. So, it must be that if the word switches membership it occurs at n or earlier. Therefore, the switch must occur before k, so k is such that for any $u, v, w, x \in \Sigma^*$, $u(vw^kx)^\omega \in L$ iff $u(vw^{k+l}x)^\omega \in L$ for all l, as required for ω-CF. ◀

▶ **Proposition 6.** *An ω-regular language L is non-uniform 1P positional*[3] *iff for all $u, y \in \Sigma^*$, $v, w, x \in \Sigma^+$:*

(1) *if $uvwx^\omega \in L$ then either $uv^\omega \in L$ or $uwx^\omega \in L$*

(2) *if $u(vwy)^\omega \in L$ then either $uvw^\omega \in L$ or $u(vy)^\omega \in L$*

Proof. ($\Longleftarrow$) Suppose we have a winning strategy from a given initial state s over a 1P arena (V, δ) with winning condition L. Since L is ω-regular, we have a finite-memory winning strategy with memory states M. Let us consider the play beginning at s generated by this strategy. There are at most $|V| \times |M|$ possible configurations of states and memory states, and clearly the play will begin looping once the same configuration is seen twice, which must happen within $|V| \times |M| + 1$ steps. Therefore this strategy will generate an ultimately periodic play $pl^\omega \subseteq \delta^\omega$ with prefix $p \subseteq \delta^*$ and infinitely looping section $l \subseteq \delta^+$. Suppose

[3] Note the results from the previous section do not necessarily apply to non-uniform 1P positional languages, as this class of languages does not require totally ordered residuals.

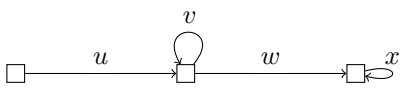
in the course of the play, we see the same state twice but select a different edge each time, meaning our strategy is not positional. i.e. the play contains for some $v \in V$, two edges $e_1, e_2 \in \delta$ such that $src(e_1) = src(e_2) = v$ but $e_1 \neq e_2$. We can use conditions (1) and (2) to modify our play to remove this violation of positionality in such a way that we still generate a winning ultimately periodic play. We will count the number of violations in a finite sequence of edges $p \in \delta^+$ as $vio(p) = \Sigma_{v \in V} count(p, v)$ where $count(p, v) = 0$ if for all edges e_1, e_2 in p such that $src(e_1) = src(e_2) = v$ then $e_1 = e_2$, and otherwise $count(p, v)$ is the number of indices i in p at which $src(p[i]) = v$. It can be seen that once $vio(p) = 0$, then for any $p_1 p_2 = p$, the ultimately periodic play $p_1 p_2^\omega$ will contain no two edges e_1, e_2 where $src(e_1) = src(e_2)$ but $e_1 \neq e_2$.

Let us first look at violations where at least one of these violating edges is in the prefix p; we will use condition (1) to fix these. Suppose our play generates the trace $uvwx^\omega \in L$. We refer to the segments of the play which generate u, v, w, and x as p_1, p_2, p_3 and p_4 respectively. Suppose $p_2[0] = e_1$ and $p_3[0] = e_2$ for two violating edges which begin from the same state as described earlier. If $uv^\omega \in L$, we can replace the play pl^ω with the winning play $p_1 p_2^\omega$, which generates the trace uv^ω. Otherwise, $uwx^\omega \in L$, and we replace pl^ω with $p_1 p_3 p_4^\omega$, which generates uwx^ω. Note that in the case where one violation is in the prefix (say, at index i of p) and one violation is in the loop (say, at index j of l) then by following this process we can reach $p[0, i-1]l[j, |l|-1]l^\omega$, for which vio does not necessarily decrease. However, this is equal to $p[0, i-1](l[j, |l|-1]l[0, j-1])^\omega$, and since every edge in $p[0, i-1]l[j, |l|-1]l[0, j-1]$ is contained within pl the same or fewer number of times and with at least one violation removed, vio decreases for this play.

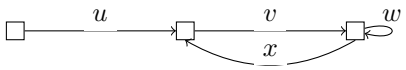
For violations entirely within l, we must alter the strategy to remove this at every instance of l within pl^ω. Therefore, we will require condition (2). Suppose our play generates a trace $u(vwx)^\omega \in L$, where $l = l_1 l_2 l_3$ st. l_1 generates v, l_2 generates w, l_3 generates x, and l_2 and l_3 begin on the same state but select different edges. Suppose $uvw^\omega \in L$; then can replace pl^ω with $pl_1 l_2^\omega$. Otherwise, $u(vx)^\omega \in L$ and we can replace pl^ω with $p(l_1 l_3)^\omega$.

Each application of these conditions replaces our play pl^ω with a new ultimately periodic play $p'l'^\omega$. Let us denote by *cycle* a sequence of edges $e_1 \dots e_n$ such that $src(e_1) = tgt(e_n)$. It can be seen that $p'l'^\omega$ is a valid play on our arena, since we have either fixed our play to end on a cycle which already existed in our original play, or we have cut out a cycle which existed in our original play. Additionally, it can be seen that $vio(p'l') < vio(pl)$, i.e. $p'l'$ has strictly fewer violations of positionality than pl, and moreover $|p'l'| < |pl|$. Since pl is finite, there are finitely many violations of positionality within pl, and once pl contains no violations of positionality then pl^ω contains no violations of positionality. Therefore, we can remove all violations of positionality with a finite number of applications of these conditions, leaving us with a play where a given state is always followed by the same edge. From this it is simple to construct a positional strategy which generates this play from the same initial state.

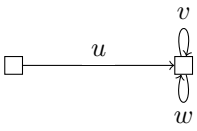
($\Longrightarrow$) Let us take an ω-regular language L which is non-uniform 1P positional. Suppose for some $uvwx^\omega \in L$, neither $uv^\omega \in L$ nor $uwx^\omega \in L$. Then, L would not be positional in the following game, where P1 controls all nodes:



Suppose for some $u(vwx)^\omega \in L$, neither $uvw^\omega \in L$ nor $u(vx)^\omega \in L$. Then, L would not be positional in the following game where P1 controls all nodes:



Additionally, in the case where x is empty, the following game provides a counterexample:



◀

▶ **Proposition 8.** *An ω-regular language L is uniform 1P positional iff it has totally ordered residuals and satisfies (1) and (2) from Proposition 6.*

Proof. For the left-to-right direction, recall that we require totally ordered residuals for positionality.

For the right-to-left direction, suppose we have a finite-memory strategy over a 1P arena, which is winning from states $W \subseteq V$. For each $w \in W$, the strategy generates an ultimately periodic play $\varsigma_w = p_w l_w^\omega$. We will modify these plays so that whenever a state is reached at multiple different points (possibly across different plays) the same edge is selected each time.

We will change the way we count violations of positionality from Proposition 6. This time, given a set of set of plays $P \subseteq \delta^\omega$, the violation count of a state $v \in V$ is $count(P, v) = |\{e \in \delta \mid src(e) = v, e = p[i] \text{ for some } i \in \mathbb{N} \text{ and some } p \in P\}|$. In other words, the violation count of v in P is the number of distinct edges across any play in P which are outgoing from v. Once $count(P, v) = 1$ for all $v \in V$, we can easily construct a positional strategy which generates all the traces in P. We will perform two different operations to lower the violation count of a given set of ultimately periodic plays P:

1. Remove violations along a single play ς_w using the method outlined in Proposition 6.
2. Remove violations across multiple plays using totally ordered residuals.

We will expand on the second operation. Let us suppose we have first applied operation (1) to remove all violations along each play - so in a given play, for every state $v \in V$, there is a single distinct outgoing edge from v that features in the play. However, the outgoing v edge in one play may not be equal to the outgoing v edge in another play. We will first select some state $v \in V$ which appears across multiple plays but is not followed by the same edge each time. We will split each play ς_w containing v into $\varsigma_w = \varsigma_{w,1}\varsigma_{w,2}$ where $\varsigma_{w,2}$ begins at the first occurrence of v in the play. Let $\lambda_{w,1}, \lambda_{w,2}$ be the words generated by $\varsigma_{w,1}$ and $\varsigma_{w,2}$ respectively. Since L has totally ordered residuals, there is some $w \in W$ such that for all other $w' \in W$, $R_L(\lambda_{w',1}) \subseteq R_L(\lambda_{w,1})$, which means that for each w', $\lambda_{w',1}\lambda_{w,2} \in L$. Therefore, if we replace each $\varsigma_{w'}$ with $\varsigma_{w',1}\varsigma_{w,2}$ for this maximal w, each play is still winning. It can also be seen that $\varsigma_{w,2}$ only contains a single distinct edge outgoing from v, namely the edge at $\varsigma_{w,2}[0]$. Since we did this at the first instance of an outgoing v edge across every play, every edge e with $src(e) = v$ across all plays is now equal to $\varsigma_{w,2}[0]$. Additionally, every play is still ultimately periodic. We will call this new set of plays P. This means our violation count for v is $count(P, v) = 1$. This operation may introduce violations along the plays for some states, in terms of the count used in Proposition 6. However, note that further applications of this operation and operation (1) will preserve the fact that every instance of v is followed by the same outgoing edge, since none of these operations add new edges that did not already exist in some play. So, once we have that $count(P, v) = 1$ for some v, this cannot be increased by operations (1) or (2). Therefore, once we repeat this process for

each $v \in V$ of applying operation (1) along all plays and then operation (2) to set all edges outgoing from v to the same choice, we are left with a set of winning plays from each $w \in W$ where a given state is always followed by the same edge. From this it is simple to construct a positional strategy which generates these plays. ◀