

# Positional Properties in Temporal Logic

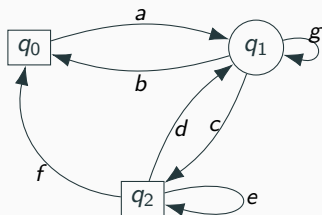
---

Jessica Newman<sup>1</sup>   Benjamin Plummer<sup>1,2</sup>

<sup>1</sup>University of Southampton

<sup>2</sup>University of Birmingham

# Motivation

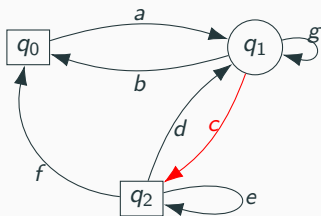


$$W \subseteq \Sigma^\omega$$

$$G = (V_1, V_2, \delta, W)$$

- Finding whether a player has a winning strategy in a 2 player game is very hard! ☹
- If we enforce bounded strategies it is easier.
- There has been a lot of recent progress on when objectives only require bounded memory.

# Memory Requirements



$$W \subseteq \Sigma^\omega$$

$$G = (V_1, V_2, \delta, W)$$

- Especially when an objective is *positional*:
- i.e. P1 has winning memoryful strategy  $\implies$  P1 has winning **positional strategy**.
- $\sigma : V_1 \rightarrow \delta$
- e.g. Parity, Büchi, Rabin.
- Note: by ‘positional’ we mean *Player 1* only requires positional strategies.

- Idea: restrict the domain of objectives to  *$\omega$ -regular positional properties*.
- c.f.
  - GR1 synthesis,
  - synthesis of safety properties,
  - **tractable fragments of temporal logics** e.g. ATL
- Couple of problems to sort out first, especially before we can apply to temporal logics...

# Problem 1

For  $\omega$ -regular objectives, there are characterisations of positionality over *edge-labelled games*. (Casares & Ohlmann 2026)

Temporal logics are usually over **state-labelled models**...

Different to positionality over edge-labelled games! So we need another characterisation.

$$(ab)^\omega$$

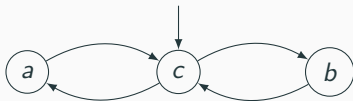


## Problem 2

How to actually generate/specify positional objectives?

It would be nice to build positional properties syntactically, but they do not compose well:

e.g.  $Fa \wedge Fb$



Our goal is to answer these two problems!

## **Characterising Positionality**

# Characterising Positionality

We will provide closure conditions on our language which guarantee positionality.

Memoryful winning strategies may visit same state twice and choose different actions.

Idea: Ensure we have enough traces in our language to ‘repair’ positionality violations in winning plays.

We will start by thinking about 1-player games.

(Edge-labelled analogue of these conditions was discovered concurrently in Colcombet & Idir 2026)

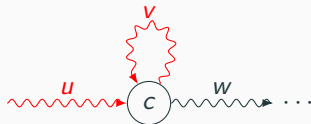
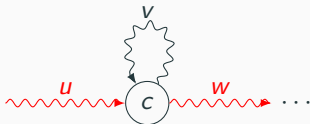
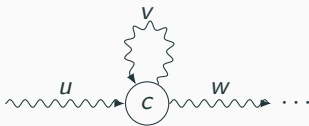
## Theorem

*An  $\omega$ -regular language  $L$  is positional over 1P state-labelled games iff the following conditions hold for all  $u, y \in \Sigma^*$ ,  $v, w, x, z \in \Sigma^+$ , where  $v[0] = w[0]$ :*

- (1) if  $uvw x^\omega \in L$  then either  $uv^\omega \in L$  or  $uw x^\omega \in L$*
- (2) if  $u(vw)^\omega \in L$  then either  $uv^\omega \in L$  or  $uw^\omega \in L$*
- (3) if  $uvy^\omega \in L$  and  $xwz^\omega \in L$  then either  $xvy^\omega \in L$  or  $uwz^\omega \in L$*

# Cutting and Looping

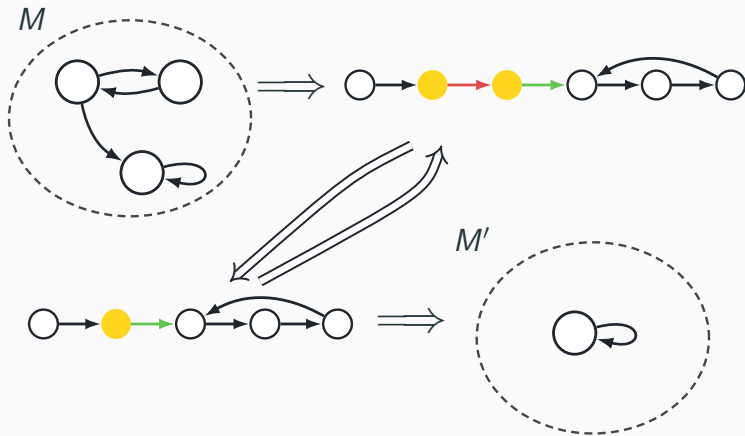
$$ucvcw \in L \implies ucw \in L \text{ or } u(cv)^\omega \in L$$



# 1-Player Positionality

With these conditions we can repair positionality violations along a linear play.

In a 1-player game, strategies generate such plays:

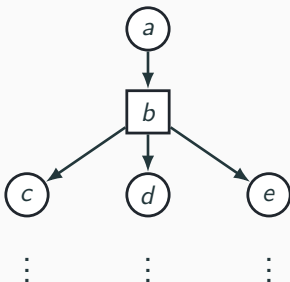


# 1-to-2 Player Lift

These conditions actually also characterise positionality over 2-player games! (1-to-2 player lift)

In 2-player games, strategies generate *trees*.

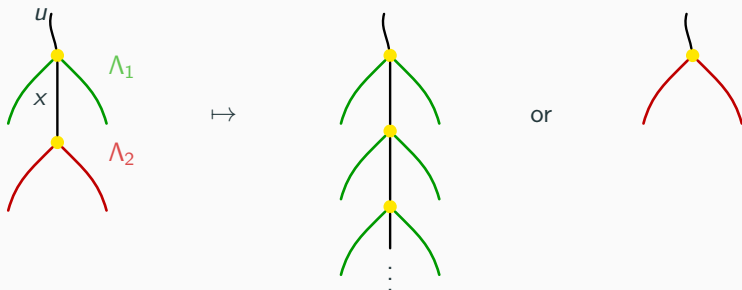
To lift these conditions to 2 player games, we need to derive conditions on trees.



# Tree conditions 1

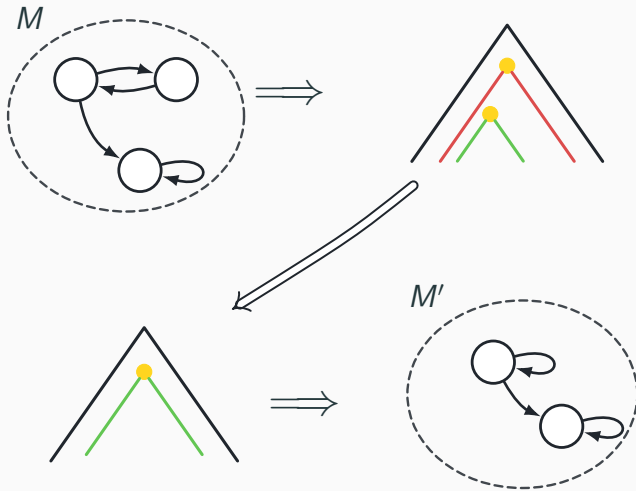
Suppose  $\Lambda_1, \Lambda_2 \subseteq c\Sigma^\omega$ ,  $x\Lambda_2 \subseteq \Lambda_1$ :

$$u\Lambda_1 \subseteq L \implies ux^n\Lambda_1 \subseteq L \text{ and } ux^\omega \in L \\ \text{or } u\Lambda_2 \subseteq L.$$



# Characterising Positionality

With these we can repair positionality violations on trees.



## Theorem

*An  $\omega$ -regular language  $L$  is state-labelled positional iff the following conditions hold for all  $u, y \in \Sigma^*$ ,  $v, w, x, z \in \Sigma^+$ , where  $v[0] = w[0]$ :*

- (1) if  $uvw x^\omega \in L$  then either  $uv^\omega \in L$  or  $uw x^\omega \in L$*
- (2) if  $u(vw)^\omega \in L$  then either  $uv^\omega \in L$  or  $uw^\omega \in L$*
- (3) if  $uvy^\omega \in L$  and  $xwz^\omega \in L$  then either  $xvy^\omega \in L$  or  $uwz^\omega \in L$*

# Characterising Positionality

$$u \in \Sigma^*, w \in \Sigma^\omega, uw \in L \iff w \in L$$

For *prefix-independent* languages, conditions (1) and (3) are met trivially:

## Proposition

A prefix-independent  $\omega$ -regular language  $L$  is state-labelled positional iff, where  $v[0] = w[0]$ ,

- $u(vw)^\omega \in L$  then either  $uv^\omega \in L$  or  $uw^\omega \in L$

## **Specifying Positionality**

The previous conditions enforce that  $L$  is aperiodic, so:

**Theorem**

*All positional  $\omega$ -regular objectives are expressible in LTL*

- How to specify properties concretely?
- One candidate: classes of positional properties closed under boolean operations.
- Our conditions have an algebraic flavour, so use Eilenberg-type correspondences:

$\infty$ -varieties  
(classes of languages of infinite words  
closed under Boolean operations,  
quotients, and inverse morphisms)



varieties of  $\omega$ -semigroups  
(classes of finite  $\omega$ -semigroups  
closed under products,  
subsemigroups, and quotients)

It's known that positional prefix-independent languages are closed under union.

## Proposition

There is no non-trivial  $\infty$ -variety of prefix-independent positional languages

The only prefix-independent positional class closed under intersection, union, and complement is the one containing  $\Sigma^\omega$  and  $\emptyset$ .

If we drop complement, we can find e.g. classes of the form:

$$FG(\Sigma_1) \vee \dots \vee FG(\Sigma_n)$$

We can also generate positional fragments of ATL\*.

e.g.

$$\langle C \rangle GF\varphi \wedge FG\psi$$

$$\langle C \rangle G(\psi U\varphi)$$

- Model checking positional properties is in PSPACE
- Model checking bipositional properties is in  $\Sigma_2^P$

## **Conclusion**

# Conclusion

- We have given a language-theoretic characterisation of  $\omega$ -regular state-labelled positional properties.
  - We can apply this characterisation to obtain results on how we can specify positional properties.
- 
- Further refine where positional  $\omega$ -regular properties are located.
  - Further investigate classes closed under union/intersection.

**Thank you!**



A. Casares and P. Ohlmann.

**Positional  $\omega$ -regular languages.**

*TheoretiCS*, Volume 5, Feb 2026.

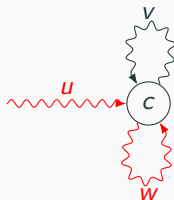
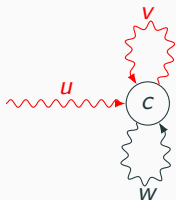
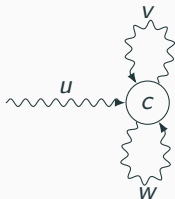


T. Colcombet and O. Idir.

**An algebraic characterisation of eve-positional languages,  
2026.**

# Cutting and Looping

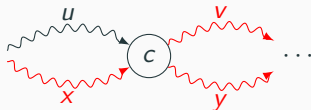
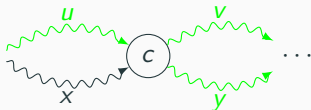
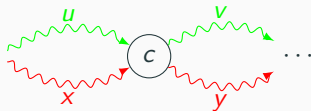
$$u(cvcw)^\omega \in L \implies u(cv)^\omega \in L \text{ or } u(cw)^\omega \in L$$



# Totally Ordered Residuals

- Note:  $u^{-1}L = \{w \in \Sigma^\omega \mid uw \in L\}$

$$uc^{-1}L \subseteq xc^{-1}L \text{ or } uc^{-1}L \supseteq xc^{-1}L$$

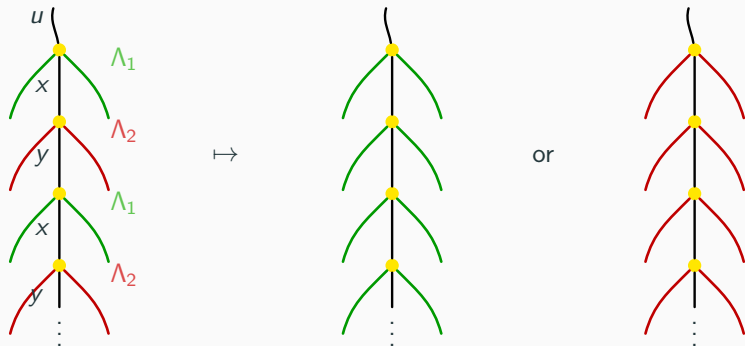


## Tree Conditions 2

Suppose  $\Lambda_1, \Lambda_2 \subseteq c\Sigma^\omega$ ,  $x\Lambda_2 \subseteq \Lambda_1$ ,  $y\Lambda_1 \subseteq \Lambda_2$ :

$$u\Lambda_1 \subseteq L \implies ux^n\Lambda_1 \subseteq L \text{ and } ux^\omega \in L$$

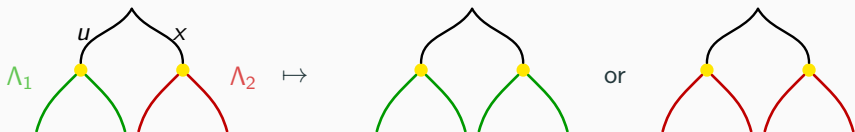
$$\text{or } uy^n\Lambda_2 \subseteq L \text{ and } uy^\omega \in L.$$



# Tree Conditions 3

Suppose  $\Lambda_1, \Lambda_2 \subseteq c\Sigma^\omega$ :

$$u\Lambda_1 \subseteq L \text{ and } x\Lambda_2 \subseteq L \implies x\Lambda_1 \subseteq L \text{ or } u\Lambda_2 \subseteq L$$



$w \approx_L w'$  iff for all  $u, v_1, v_2 \in \Sigma^*$ ,  $v_1 w v_2 u^\omega \in L$  iff  $v_1 w' v_2 u^\omega \in L$  and  $u(v_1 w v_2)^\omega \in L$  iff  $u(v_1 w' v_2)^\omega$

The previous conditions enforce that  $\Sigma^* / \approx_L$  is aperiodic