How to Play Optimally for Regular Objectives?

Patricia Bouyer[1], Nathanaël Fijalkow[2],
Mickael Randour[3], **Pierre Vandenhove**[1,3]

[1]Université Paris-Saclay, CNRS, ENS Paris-Saclay, LMF, France

[2]LaBRI, Université de Bordeaux, France

[3]F.R.S.-FNRS & UMONS – Université de Mons, Belgium

February 17, 2023 – GT Informel CDS & MCS, LMF

Laboratoire Méthodes Formelles

FREEDOM TO RESEARCH

UMONS Université de Mons

Outline

Synthesis problem

Synthesizing **controllers** for **reactive systems** with an **objective**.
Systems and their environment modeled with **zero-sum games**.

Strategy complexity

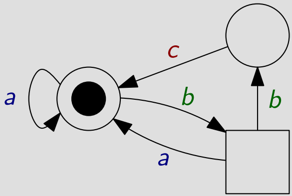
Given an objective, what are the **smallest** optimal controllers?
$\rightsquigarrow$ What is the *smallest automatic structure* remembering **sufficient information** to make **optimal decisions**?

Results

Characterization of automatic structures for *regular objectives*;
computational complexity of finding small structures.

Games

Zero-sum turn-based games on graphs



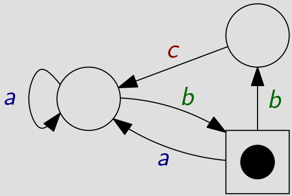
- $C = \{a, b, c\}$, $\mathcal{A} = (V_1, V_2, E)$.
- Two players $\mathcal{P}_1$ ($\bigcirc$) and $\mathcal{P}_2$ ($\square$)

Strategies

A **strategy** of a player is a function $\sigma\colon E^* \to E$.
A strategy σ of $\mathcal{P}_1$ is **winning for** W **from** $v \in V$ if all infinite paths from v *consistent with* σ induce an infinite word in W.

Games

Zero-sum turn-based games on graphs



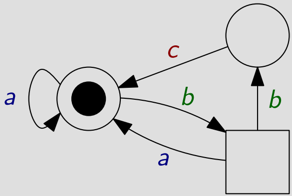
- $C = \{a, b, c\}$, $\mathcal{A} = (V_1, V_2, E)$.
- Two players $\mathcal{P}_1$ ($\bigcirc$) and $\mathcal{P}_2$ ($\square$) generate an infinite word $w = b$

Strategies

A **strategy** of a player is a function $\sigma \colon E^* \to E$.
A strategy σ of $\mathcal{P}_1$ is **winning for** W **from** $v \in V$ if all infinite paths from v *consistent with* σ induce an infinite word in W.

Games

Zero-sum turn-based games on graphs



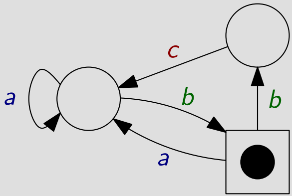
- $C = \{a, b, c\}$, $\mathcal{A} = (V_1, V_2, E)$.
- Two players $\mathcal{P}_1$ ($\bigcirc$) and $\mathcal{P}_2$ ($\square$) generate an infinite word $w = ba$

Strategies

A **strategy** of a player is a function $\sigma\colon E^* \to E$.
A strategy σ of $\mathcal{P}_1$ is **winning for** W **from** $v \in V$ if all infinite paths from v *consistent with* σ induce an infinite word in W.

Games

Zero-sum turn-based games on graphs



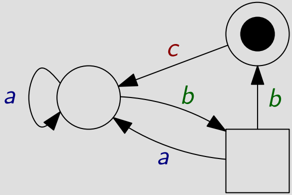
- $C = \{a, b, c\}$, $\mathcal{A} = (V_1, V_2, E)$.
- Two players $\mathcal{P}_1$ ($\bigcirc$) and $\mathcal{P}_2$ ($\square$) generate an infinite word $w = bab$

Strategies

A **strategy** of a player is a function $\sigma \colon E^* \to E$.
A strategy σ of $\mathcal{P}_1$ is **winning for** W **from** $v \in V$ if all infinite paths from v *consistent with* σ induce an infinite word in W.

Games

Zero-sum turn-based games on graphs



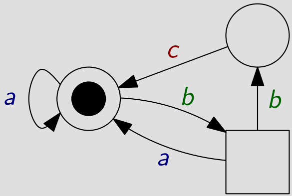
- $C = \{a, b, c\}$, $\mathcal{A} = (V_1, V_2, E)$.
- Two players $\mathcal{P}_1$ ($\bigcirc$) and $\mathcal{P}_2$ ($\square$) generate an infinite word $w = babb$

Strategies

A **strategy** of a player is a function $\sigma\colon E^* \to E$.
A strategy σ of $\mathcal{P}_1$ is **winning for** W **from** $v \in V$ if all infinite paths from v *consistent with* σ induce an infinite word in W.

Games

Zero-sum turn-based games on graphs



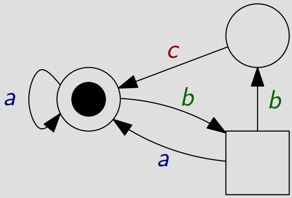
- $C = \{a, b, c\}$, $\mathcal{A} = (V_1, V_2, E)$.
- Two players $\mathcal{P}_1$ ($\bigcirc$) and $\mathcal{P}_2$ ($\square$) generate an infinite word $w = babbc \ldots \in C^\omega$.

Strategies

A **strategy** of a player is a function $\sigma \colon E^* \to E$.
A strategy σ of $\mathcal{P}_1$ is **winning for** W **from** $v \in V$ if all infinite paths from v *consistent with* σ induce an infinite word in W.

Games

Zero-sum turn-based games on graphs



- $C = \{a, b, c\}$, $\mathcal{A} = (V_1, V_2, E)$.
- Two players $\mathcal{P}_1$ ($\bigcirc$) and $\mathcal{P}_2$ ($\square$) generate an infinite word $w = babbc \ldots \in C^\omega$.
- **Objective** of $\mathcal{P}_1$ is a set $W \subseteq C^\omega$.
- **Zero-sum**: objective of $\mathcal{P}_2$ is $C^\omega \setminus W$.

Strategies

A **strategy** of a player is a function $\sigma\colon E^* \to E$.
A strategy σ of $\mathcal{P}_1$ is **winning for** W **from** $v \in V$ if all infinite paths from v *consistent with* σ induce an infinite word in W.

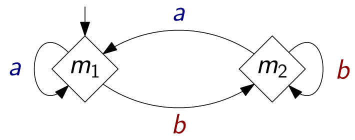
Representations of a strategy

In general, a strategy $\sigma\colon E^* \to E$ has an **infinite** representation.
For synthesis, we like when it has a **finite** representation with a **computable** size. Usual finite representation:

Memory structure

Memory structure $(M, m_{\mathsf{init}}, \alpha_{\mathsf{upd}})$: finite set of states M, initial state m_{init}, update function $\alpha_{\mathsf{upd}}\colon M \times C \to M$.

Ex.: remember whether a or b was last played (**not yet a strategy!**):

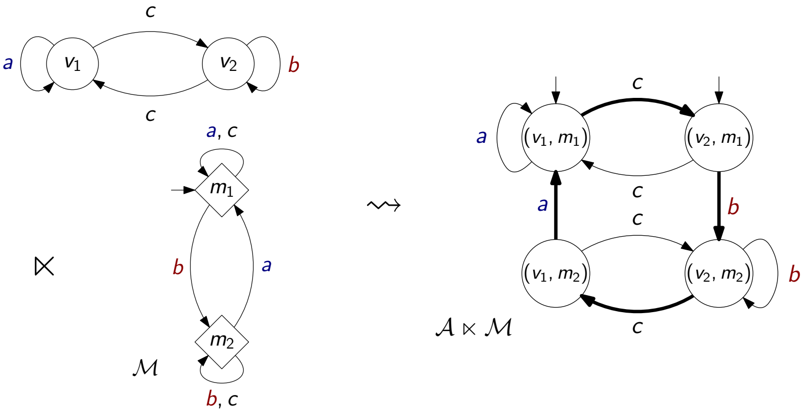


Given an arena $\mathcal{A} = (V_1, V_2, E)$: *next-action function* $\alpha_{\mathsf{nxt}}\colon V_i \times M \to E$.

Finite memory ≈ no memory in the product

Memory $\mathcal{M}$ in $\mathcal{A}$ ≈ no memory in arena $\mathcal{A} \ltimes \mathcal{M}$.

If $C = \{a, b, c\}$,
$W = \{w \in C^\omega \mid a \text{ is seen } \infty\text{ly often and } b \text{ is seen } \infty\text{ly often}\}$:



ω-regular objectives

The *ω-regular objectives* are the ones that can be expressed with **ω-regular expressions**, or with **deterministic Muller automata**.

Examples with $C = \{a, b\}$:

- $W = b^*ab^*aC^\omega$;
- $W = (b^*a)^\omega$.

Theorem (Büchi, Landweber, 1969)[1]

All **ω-regular objectives** admit **finite-memory winning strategies** in all arenas.

[1] Büchi and Landweber, "Definability in the Monadic Second-Order Theory of Successor", 1969.

Well-studied case: *Muller conditions*

For $\mathcal{F} \subseteq 2^C$, *objective* Muller($\mathcal{F}$) is the set of words whose set of colors seen infinitely often is in $\mathcal{F}$.

Examples with $C = \{a, b\}$:

- $\text{Muller}(\{\{a\}, \{a, b\}\}) = (b^* a)^\omega$,
- $\text{Muller}(\{\{a, b\}\}) = (b^* a)^\omega \cap (a^* b)^\omega$.

Memory requirements of Muller conditions

- First upper bound of size $\mathcal{O}(|C|!)$ in 1982 (*later appearance record*);[2]
- Many works about specific cases;[3,4]
- **Characterization** of precise memory requirements and **algorithm** to compute them in 1997 ([DJW97][5]).

---

[2] Gurevich and Harrington, "Trees, Automata, and Games", 1982.

[3] Emerson and Jutla, "Tree Automata, Mu-Calculus and Determinacy (Extended Abstract)", 1991.

[4] Klarlund, "Progress Measures, Immediate Determinacy, and a Subset Construction for Tree Automata", 1994.

[5] Dziembowski, Jurdziński, and Walukiewicz, "How Much Memory is Needed to Win Infinite Games?", 1997.

Is that it?

We have:

1. that ω-regular objectives can be represented by a **deterministic automaton using a Muller acceptance condition**;
2. a complete understanding of the **memory requirements of Muller conditions**.

Does this settle the question of the memory requirements of all ω-regular objectives?

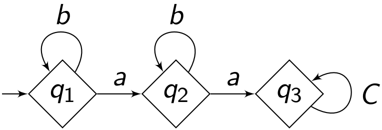
Has been quoted as such,[6] but **not the case** (it is only an upper bound)!

[6] In *Handbook of Model Checking* (Bloem, Chatterjee, and Jobstmann, "Graph Games and Reactive Synthesis", 2018): "The results of Dziembowski et al. [80] give precise memory requirements for strategies in 2-player games with ω-regular objectives".

Why only an upper bound?

Let $C = \{a, b\}$, $W = b^*ab^*aC^\omega$ ($\approx$ *seeing a two or more times*).
How to use results about **Muller conditions**?

W is not directly a Muller condition Muller($\mathcal{F}$) with $\mathcal{F} \subseteq 2^C$
$\rightsquigarrow$ needs an **automaton structure**.



$\rightsquigarrow W = \text{Muller}(\{\{q_3\}\})$.

Using [DJW97],[7] we need 1 memory state...

... **after** augmenting the arenas with the automaton,

so **upper bound of 3 states of memory**.

But 1 **memory state** suffices for winning strategies!

[7] Dziembowski, Jurdziński, and Walukiewicz, "How Much Memory is Needed to Win Infinite Games?", 1997.

Orthogonal quest: **regular** objectives

How to go further?

Study the memory requirements of ω-regular objectives with **non-trivial automaton structures**.

We consider the "simplest" ones.

Regular objectives

- A **regular reachability objective** is a set LC^ω with $L \subseteq C^*$ regular.
- A **regular safety objective** is a set $C^\omega \setminus LC^\omega$.

- A player wants to realize a word in L, the other wants to prevent it.
- Expressible as standard **deterministic finite automata**.
- *Special cases of open and closed sets, at the first level of the Borel hierarchy*.

Question

Memory requirements of regular objectives

Characterize the memory structures that suffice to make optimal decisions for regular objectives **in any arena**. Compute **minimal** ones.

Comparing words

Let $W \subseteq C^\omega$ be an objective.

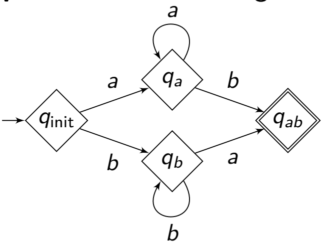
Winning continuations

For $x \in C^*$, $x^{-1}W = \{w \in C^\omega \mid xw \in W\}$.

For $x, y \in C^*$,

- $x \sim_W y$ if $x^{-1}W = y^{-1}W$ ($\approx$ *Myhill-Nerode equivalence relation*),
- $x \preceq_W y$ if $x^{-1}W \subseteq y^{-1}W$ (preorder).

Example: let W be the regular **safety** objective induced by this DFA.



$\varepsilon^{-1}W = W$, $a^{-1}W = \{a^\omega\}$,
$b^{-1}W = \{b^\omega\}$, $(ab)^{-1}W = \emptyset$.

E.g., $a \prec_W \varepsilon$, $ab \prec_W a$,
a and b are incomparable for $\preceq_W$.

Necessary condition for the memory

Let W be an objective.

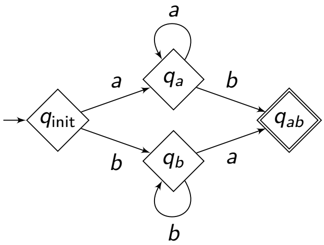
Lemma

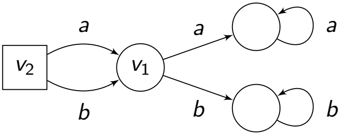
A sufficient memory structure $\mathcal{M} = (M, m_{\mathsf{init}}, \alpha_{\mathsf{upd}})$ needs to **distinguish incomparable words** (for $\preceq_W$), i.e.,

$$\text{if } x, y \in C^* \text{ are incomparable for } \preceq_W,$$
$$\text{then } \alpha^*_{\mathsf{upd}}(m_{\mathsf{init}}, x) \neq \alpha^*_{\mathsf{upd}}(m_{\mathsf{init}}, y).$$

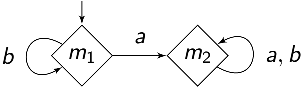
Why is it necessary?

Example of a regular **safety** objective, with a and b incomparable.





The memory structure needs to "distinguish" the incomparable a and b.
One memory that suffices:



Characterization: safety

Let W be a **regular safety objective**.

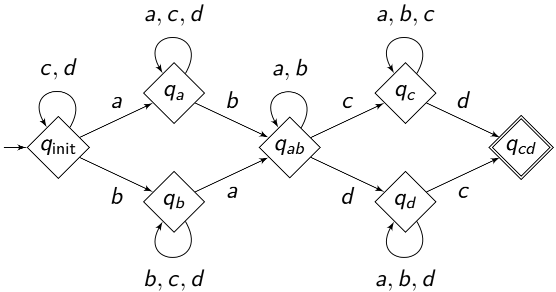
Theorem

A memory structure $\mathcal{M}$ implements winning strategies in all arenas
if and only if
$\mathcal{M}$ distinguishes incomparable words.

Question

How to find a smallest such memory structure?

More involved example (1/2)

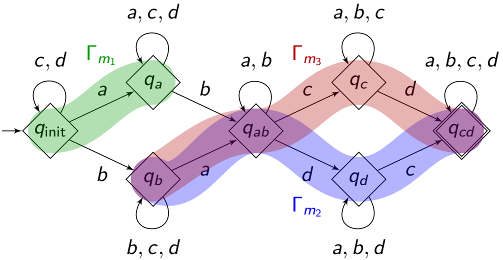


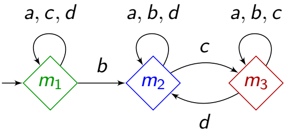
Only (q_a, q_b) and (q_c, q_d) are pairs of incomparable states.

Taking the whole automaton as a memory always works $\rightsquigarrow$ 7 states.

More involved example (2/2)

Possible to do better? **Yes**!





$\rightsquigarrow$ Combinatorial reformulation of "structure $\mathcal{M}$ distinguishing incomparable prefixes" into a **covering of the states** with good properties.

Computational complexity: safety

Decision problem

MEMORYSAFE
Input: An automaton $\mathcal{D}$ inducing the regular safety objective W and $k \in \mathbb{N}$.
Question: $\exists$ a memory structure $\mathcal{M}$ with $\leq k$ states that suffices for W?

Theorem

MEMORYSAFE is NP-complete.

Thanks to the covering reformulation,

- MEMORYSAFE is in NP;
- NP-hardness with a reduction from HAMILTONIANCYCLE.

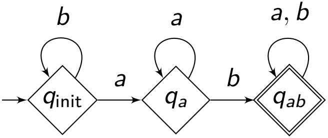
Regular reachability

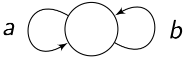
Let W be a regular **reachability** objective.

Memory structures still need to **distinguish incomparable words**.

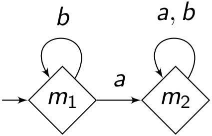
But **not sufficient**!

$W = b^* a a^* b C^\omega$ (all words are comparable):





Main idea: seeing a is necessary and *makes progress*. However, we cannot just play a to win. Word a is an *insufficient progress*.



Condition necessary for reachability

Let $W \subseteq C^\omega$ be an objective.

Necessary property

Let $\mathcal{M} = (M, m_{\mathsf{init}}, \alpha_{\mathsf{upd}})$ be a memory structure.
Memory structure $\mathcal{M}$ **distinguishes insufficient progress** if

$$\text{for all } w_1, w_2 \in C^*, \text{ if } w_1 \prec_W w_1 w_2 \text{ and } w_1(w_2)^\omega \notin W,$$
$$\text{then } \alpha^*_{\mathsf{upd}}(m_{\mathsf{init}}, w_1) \neq \alpha^*_{\mathsf{upd}}(m_{\mathsf{init}}, w_1 w_2).$$

Also necessary to implement winning strategies for **any** objective.

Characterization: reachability

Let W be a **regular reachability objective**.

Theorem

Memory structure $\mathcal{M}$ implements winning strategies in all arenas
if and only if
$\mathcal{M}$ distinguishes incomparable words **and**
$\mathcal{M}$ distinguishes insufficient progress.

Remark

Distinguishing insufficient progress is necessary for all objectives, even for regular **safety** ones...
... but there is *no insufficient progress* for regular safety objectives!

Computational complexity: reachability

Decision problem

MemoryReach
Input: An automaton $\mathcal{D}$ inducing the regular reachability objective W and $k \in \mathbb{N}$.
Question: $\exists$ a memory structure $\mathcal{M}$ with $\leq k$ states that suffices for W?

Theorem

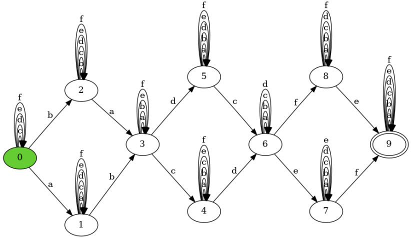
MemoryReach is NP-complete.

Needed to show that "$\mathcal{M}$ distinguishes insufficient progress" is in NP, but the same hardness proof as for MemorySafe.

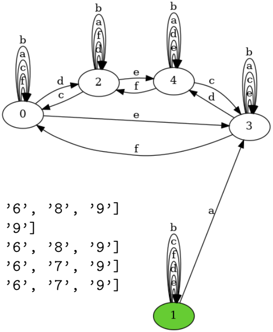
Implementation

Algorithms[8] that find minimal memory structures for regular objectives.
Simple ideas: binary search on the minimal size, encoding properties as SAT instances and use of a SAT solver.

D =




```
M = memReq.smallest_memory_safety(D)
```




```
\Gamma_0 = ['1', '3', '4', '6', '8', '9']
\Gamma_1 = ['0', '2', '7', '9']
\Gamma_2 = ['1', '3', '5', '6', '8', '9']
\Gamma_3 = ['1', '3', '4', '6', '7', '9']
\Gamma_4 = ['1', '3', '5', '6', '7', '9']
```

[8] https://github.com/pvdhove/regularMemoryRequirements

Conclusion

Summary

- Characterization of the memory structures for **regular objectives**.
- **NP-completeness** of finding small memory structures.
- Implementation using a SAT solver.

Future work

Two orthogonal directions: *Muller conditions*[9,10] and *regular objectives*.[11]

$\rightsquigarrow$ What about Muller automata, i.e., **ω-regular objectives**?

Partial results for **deterministic Büchi automata**.[12]

Thanks!

[9] Dziembowski, Jurdziński, and Walukiewicz, "How Much Memory is Needed to Win Infinite Games?", 1997.

[10] Casares, "On the Minimisation of Transition-Based Rabin Automata and the Chromatic Memory Requirements of Muller Conditions", 2022.

[11] Bouyer, Fijalkow, et al., "How to Play Optimally for Regular Objectives?", 2022.

[12] Bouyer, Casares, et al., "Half-Positional Objectives Recognized by Deterministic Büchi Automata", 2022.