Layered Automata

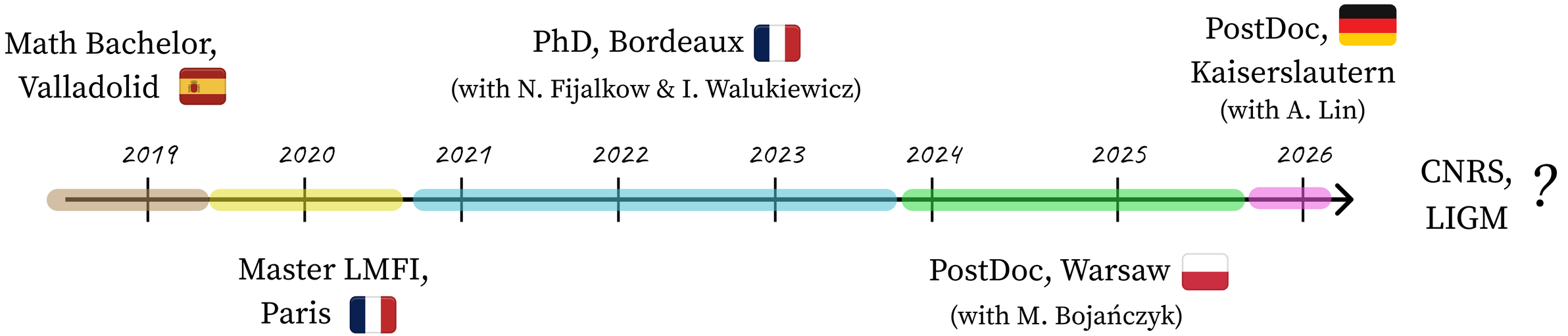
A canonical model for ω-regular languages

Antonio Casares · RPTU Kaiserslautern

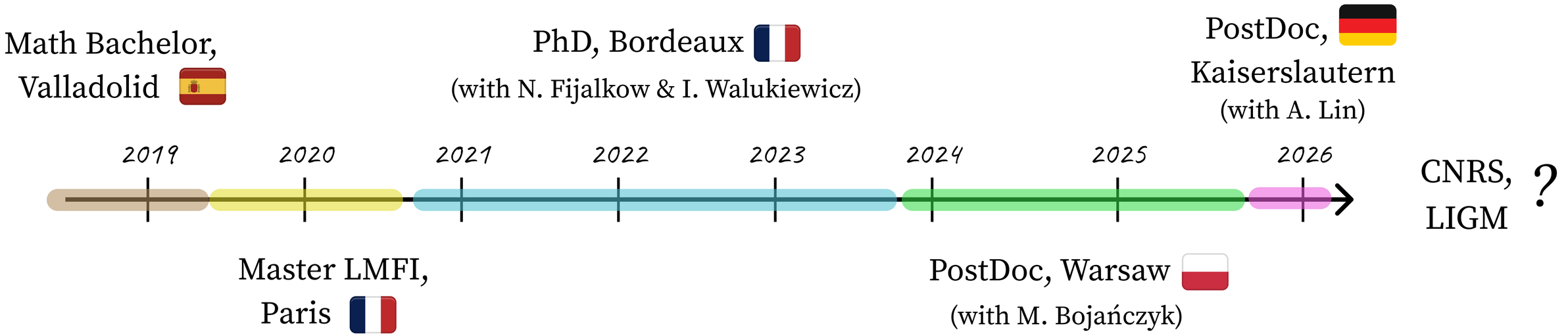
Joint work with: Christof Löding and Igor Walukiewicz

Séminaire BAAM, 4 November 2025

Who am I?


Math Bachelor,
Valladolid
PhD, Bordeaux
(with N. Fijalkow & I. Walukiewicz)
PostDoc,
Kaiserslautern
(with A. Lin)
2019
2020
2021
2022
2023
2024
2025
2026
CNRS,
LIGM
?
Master LMFI,
Paris
PostDoc, Warsaw
(with M. Bojańczyk)

Who am I?



Courses in
- Model theory
- Descriptive complexity

ω-automata

Games on graphs

Logic

Transducers

Finite model theory

Graph databases

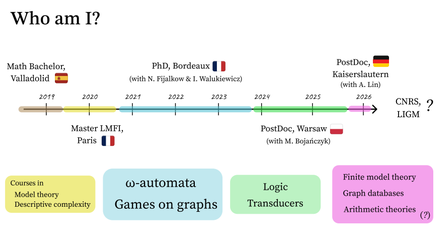
Arithmetic theories (?)

Layered Automata

A canonical model for ω-regular languages

Antonio Casares · RPTU Kaiserslautern

Joint work with: Christof Löding and Igor Walukiewicz



Séminaire BAAM, 4 November 2025

Everybody loves automata

✦ Finite automata $\longleftrightarrow$ MSO-logic *(Büchi, Elgot, Trakhtenbrot '60)*

✦ Finite automata ⟷ MSO-logic *(Büchi, Elgot, Trakhtenbrot '60)*

A minimal, canonical DFA
for each regular language

✦ Finite automata ⟷ MSO-logic *(Büchi, Elgot, Trakhtenbrot '60)*

Beautiful mathematical theory!

A minimal, canonical DFA
for each regular language

✦ Finite automata ⟷ MSO-logic *(Büchi, Elgot, Trakhtenbrot '60)*

Beautiful mathematical theory!

A minimal, canonical DFA
for each regular language

Efficiency in applications!

- Finite automata $\longleftrightarrow$ MSO-logic *(Büchi, Elgot, Trakhtenbrot '60)*

Beautiful mathematical theory!

A minimal, canonical DFA
for each regular language

Efficiency in applications!

- Minimisation in $O(n \log n)$ *(Hopcroft '71)*

- Finite automata ⟷ MSO-logic *(Büchi, Elgot, Trakhtenbrot '60)*

Beautiful mathematical theory!

A minimal, canonical DFA
for each regular language

Efficiency in applications!

- Minimisation in *O(nlogn)* *(Hopcroft '71)*
- Automata learning in PTIME *(Gold '67, Angluin 87')*

✦ Finite automata $\longleftrightarrow$ MSO-logic *(Büchi, Elgot, Trakhtenbrot '60)*

Beautiful mathematical theory!

A minimal, canonical DFA
for each regular language

Efficiency in applications!

- ✦ Minimisation in $O(n \log n)$ *(Hopcroft '71)*
- ✦ Automata learning in PTIME *(Gold '67, Angluin 87')*
- ✦ Simple characterization of FO-definability

⋮

Everybody loves automata

+ Finite automata ⟷ MSO-logic *(Büchi, Elgot, Trakhtenbrot '60)*

Beautiful mathematical theory!

A minimal, canonical DFA for each regular language

Efficiency in applications!

+ Minimisation in *O(nlogn) (Hopcroft '71)*
+ Automata learning in PTIME *(Gold '67, Angluin 87')*
+ Simple characterization of FO-definability

⋮

Everybody loves automata

+ Finite automata ⟷ MSO-logic *(Büchi, Elgot, Trakhtenbrot '60)*

Beautiful mathematical theory!

A minimal, canonical DFA for each regular language

Efficiency in applications!

+ Minimisation in *O(nlogn) (Hopcroft '71)*
+ Automata learning in PTIME *(Gold '67, Angluin 87')*
+ Simple characterization of FO-definability

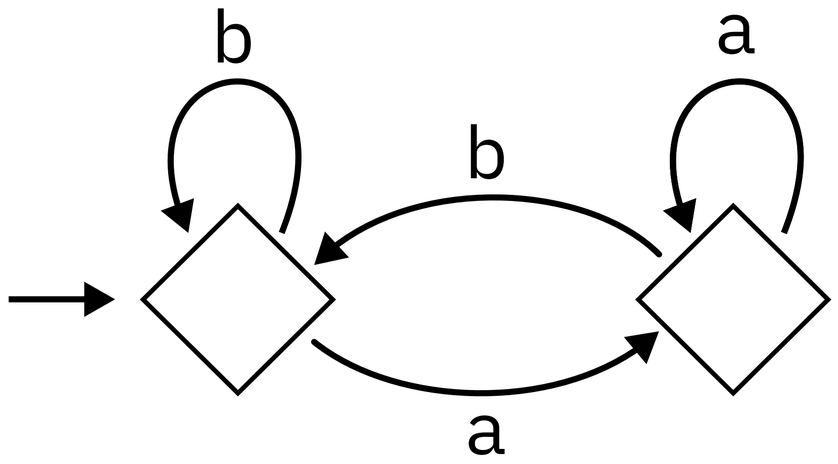
⋮

Let’s generalize to infinite words!

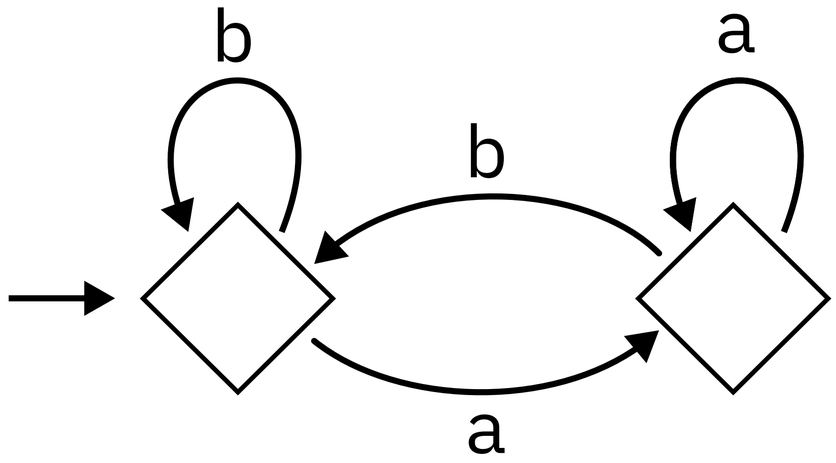
✦ ω-automata $\longleftrightarrow$ MSO-logic *(Büchi '62)*

- ω-automata $\longleftrightarrow$ MSO-logic *(Büchi '62)*
- Applications in verification and reactive synthesis

ω-automata

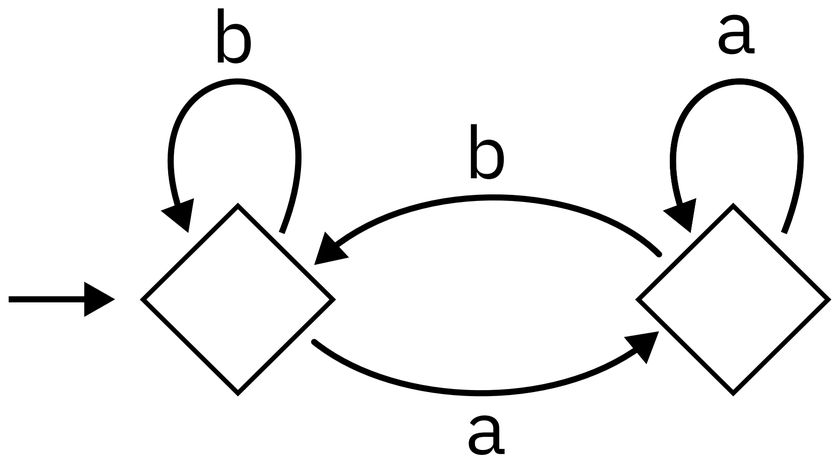


ω-automata



Input: abaababbaa... $\in \Sigma^{\omega}$

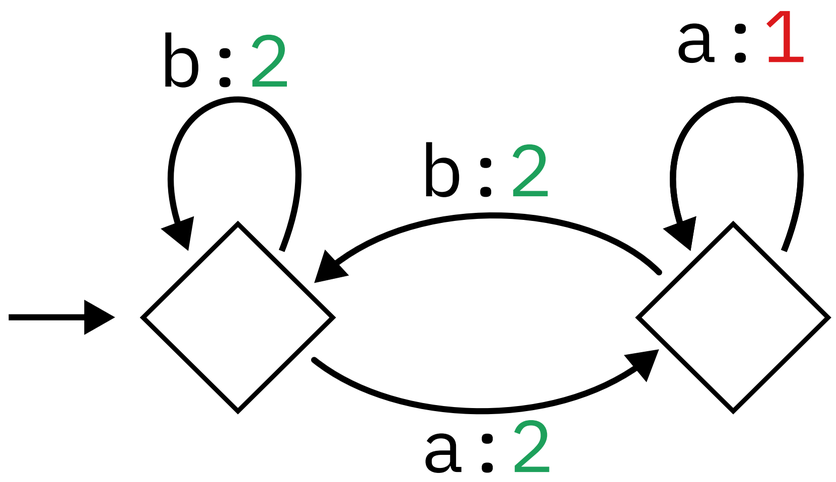
ω-automata



Acceptance?

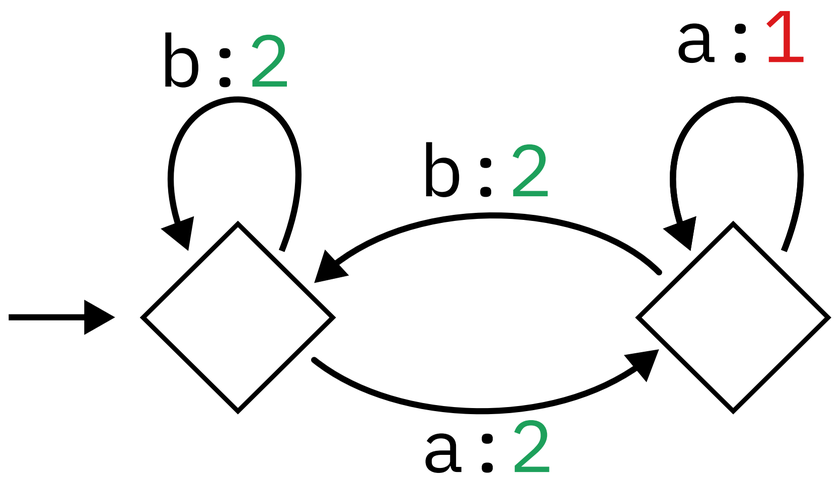
Input: abaababbbaa... $\in \Sigma^{\omega}$

Parity automata



Input: $abaababbaa\ldots \in \Sigma^{\omega}$

Parity automata

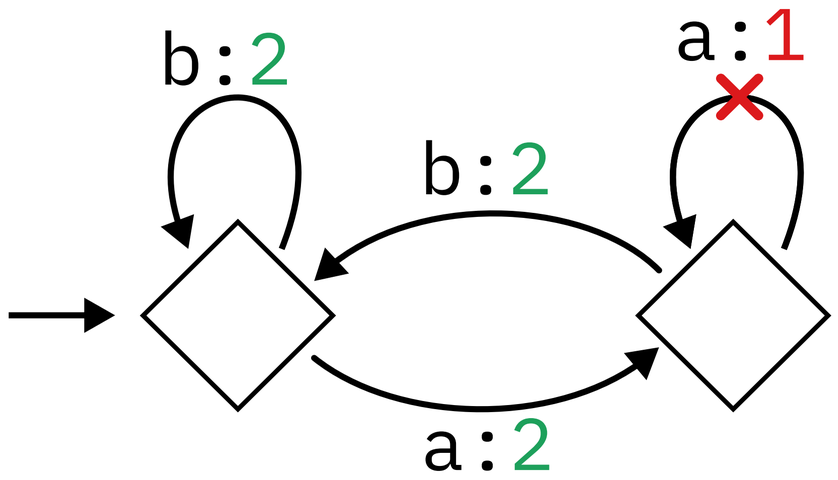


Run accepting if

min {*priorities seen infinitely often*}
is even

Input: abaababbaa... $\in \Sigma^{\omega}$

Parity automata



Run accepting if

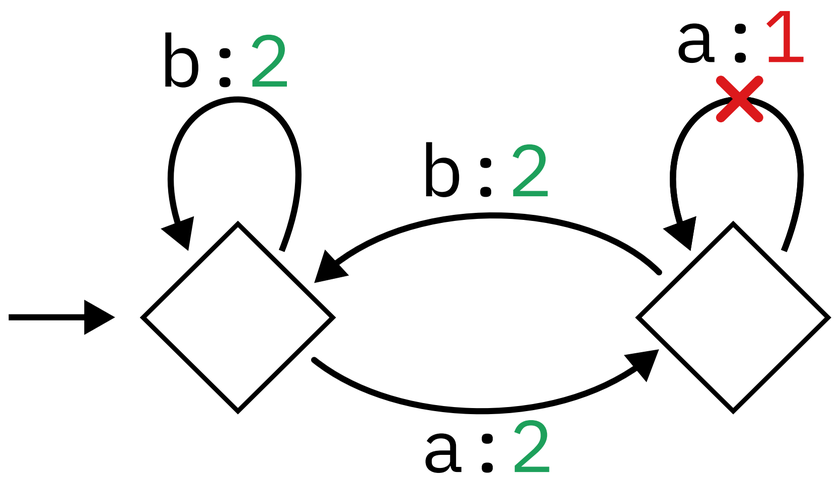
min {*priorities seen infinitely often*}
is even

Input: `abaababbaa...` $\in \Sigma^{\omega}$

$L(A)$ = `Finitely Often` `` `aa` ``

Parity automata

Acceptance on transitions!



Run accepting if

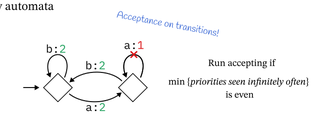
min {*priorities seen infinitely often*}
is even

Input: `abaababbbaa...` $\in \Sigma^\omega$

$L(A)$ = `Finitely Often` `` `aa` ``

- ω-automata ⟷ MSO-logic *(Büchi '62)*
- Applications in verification and reactive synthesis

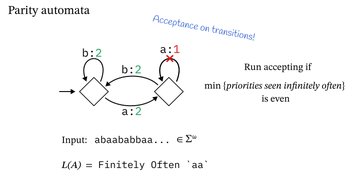
Parity automata



Input: abaababbaa... $\in \Sigma^*$

L(A) = Finitely Often `aa`

- ω-automata ⟷ MSO-logic *(Büchi '62)*
- Applications in verification and reactive synthesis



No unique minimal deterministic automata

Myhill-Nerode equivalence relation:

$$u \sim v \quad \text{if} \quad \text{for all } w \in \Sigma^{\omega}, \quad uw \in L \iff vw \in L$$

Myhill-Nerode equivalence relation:

$$u \sim v \quad \text{if} \quad \text{for all } w \in \Sigma^\omega, \;\; uw \in L \iff vw \in L$$

$$L = \texttt{Finitely Often `aa`}$$

Myhill-Nerode equivalence relation:

$$u \sim v \quad \text{if} \quad \text{for all } w \in \Sigma^{\omega}, \;\; uw \in L \iff vw \in L$$

$L =$ `Finitely Often \`aa\``

Has only one residual!

Myhill-Nerode equivalence relation:

$$u \sim v \quad \text{if} \quad \text{for all } w \in \Sigma^\omega, \;\; uw \in L \iff vw \in L$$

$L =$ `Finitely Often` `aa`

Has only one residual!

No 1-state automaton can recognize L

Myhill-Nerode equivalence relation:

$$u \sim v \quad \text{if} \quad \text{for all } w \in \Sigma^\omega, \quad uw \in L \iff vw \in L$$

$L =$ `Finitely Often aa` **Has only one residual!**

No 1-state automaton can recognize L

The automaton of residuals is not enough

Myhill-Nerode equivalence relation:

$$u \sim v \quad \text{if} \quad \text{for all } w \in \Sigma^\omega, \quad uw \in L \iff vw \in L$$

$L =$ `Finitely Often `aa`` **Has only one residual!**

No 1-state automaton can recognize L

The automaton of residuals is not enough

We may need several language-equivalent states

- ω-automata ⟷ MSO-logic *(Büchi '62)*
- Applications in verification and reactive synthesis

Parity automata

Run accepting if
min {*priorities seen infinitely often*}
is even

Input: `abaababbaa...` $\in \Sigma^\omega$

$L(A)$ = `Finitely Often aa`

Myhill-Nerode equivalence relation:

$u \sim v$ if for all $w \in \Sigma^*$, $uw \in L \iff vw \in L$

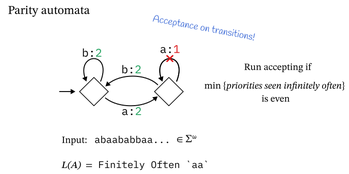
L = `Finitely Often aa` **Has only one residual!**

No 1-state automaton can recognize L

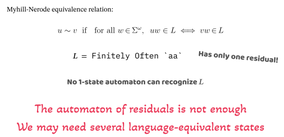
The automaton of residuals is not enough
We may need several language-equivalent states

No unique minimal deterministic automata

✦ ω-automata ⟷ MSO-logic *(Büchi '62)*



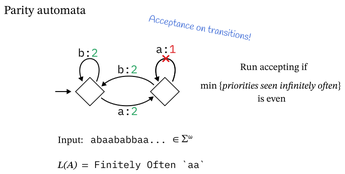
✦ Applications in verification and reactive synthesis



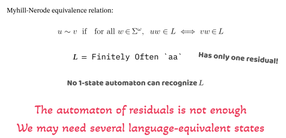
No unique minimal deterministic automata

✖ Minimisation is NP-complete *(Schewe '10, AbuRadi-Ehlers '25)*

✦ ω-automata ⟷ MSO-logic *(Büchi '62)*



✦ Applications in verification and reactive synthesis

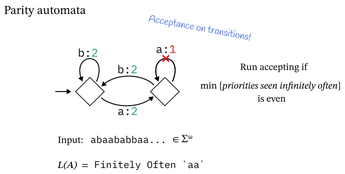


No unique minimal deterministic automata

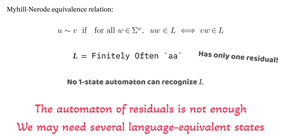
Use of ω-automata very costly in applications!

✖ Minimisation is NP-complete *(Schewe '10, AbuRadi-Ehlers '25)*

- ω-automata ⟷ MSO-logic *(Büchi '62)*



- Applications in verification and reactive synthesis

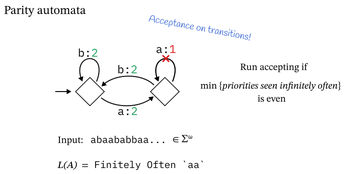


No unique minimal deterministic automata

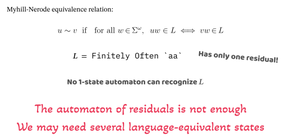
Use of ω-automata very costly in applications!

- Minimisation is NP-complete *(Schewe '10, AbuRadi-Ehlers '25)*
- Costly determinization and boolean operations

- ω-automata $\longleftrightarrow$ MSO-logic *(Büchi '62)*



- Applications in verification and reactive synthesis



No unique minimal deterministic automata

Use of ω-automata very costly in applications!

- Minimisation is NP-complete *(Schewe '10, AbuRadi-Ehlers '25)*
- Costly determinization and boolean operations
- No efficient learning of ω-automata

Everybody loves automata

- Finite automata ⟷ MSO-logic *(Büchi, Elgot, Trakhtenbrot '60)*

Beautiful mathematical theory!

A minimal, canonical DFA for each regular language

Efficiency in applications!

- Minimisation in $O(n \log n)$ *(Hopcroft '71)*
- Automata learning in PTIME *(Gold '67, Angluin 87')*
- Simple characterization of FO-definability

⋮

Let's generalize to infinite words!

- ω-automata ⟷ MSO-logic *(Büchi '62)*

- Applications in verification and reactive synthesis

No unique minimal deterministic automata

Use of ω-automata very costly in applications!

- ✗ Minimisation is NP-complete *(Schewe '10, AlurRadi-Ehlers '25)*
- ✗ Costly determinization and boolean operations
- ✗ No efficient learning of ω-automata

Everybody loves automata

+ Finite automata ⟷ MSO-logic *(Büchi, Elgot, Trakhtenbrot '60)*

Beautiful mathematical theory!

A minimal, canonical DFA for each regular language

Efficiency in applications!

+ Minimisation in $O(n\log n)$ *(Hopcroft '71)*
+ Automata learning in PTIME *(Gold '67, Angluin 87')*
+ Simple characterization of FO-definability

⋮

Let's generalize to infinite words!

+ ω-automata ⟷ MSO-logic *(Büchi '62)*

+ Applications in verification and reactive synthesis

No unique minimal deterministic automata

Use of ω-automata very costly in applications!

× Minimisation is NP-complete *(Schewe '10, AlurRadi-Ehlers '25)*
× Costly determinization and boolean operations
× No efficient learning of ω-automata

The theory of deterministic parity automata is not satisfactory...

Everybody loves automata

+ Finite automata ⟷ MSO-logic *(Büchi, Elgot, Trakhtenbrot '60)*

Beautiful mathematical theory!

A minimal, canonical DFA for each regular language

Efficiency in applications!

+ Minimisation in $O(n\log n)$ *(Hopcroft '71)*
+ Automata learning in PTIME *(Gold '67, Angluin 87')*
+ Simple characterization of FO-definability

⋮

Let's generalize to infinite words!

+ ω-automata ⟷ MSO-logic *(Büchi '62)*
+ Applications in verification and reactive synthesis

No unique minimal deterministic automata

Use of ω-automata very costly in applications!

× Minimisation is NP-complete *(Schewe '10, AlurRadi-Ehlers '25)*
× Costly determinization and boolean operations
× No efficient learning of ω-automata

The theory of deterministic parity automata is not satisfactory...

Instead, use ***Layered Automata***

Everybody loves automata

+ Finite automata ⟷ MSO-logic *(Büchi, Elgot, Trakhtenbrot '60)*

Beautiful mathematical theory!

A minimal, canonical DFA for each regular language

Efficiency in applications!

+ Minimisation in $O(n \log n)$ *(Hopcroft '71)*
+ Automata learning in PTIME *(Gold '67, Angluin 87')*
+ Simple characterization of FO-definability

⋮

Let's generalize to infinite words!

+ ω-automata ⟷ MSO-logic *(Büchi '62)*

+ Applications in verification and reactive synthesis

No unique minimal deterministic automata

Use of ω-automata very costly in applications!

× Minimisation is NP-complete *(Schewe '10, AlurRadi-Ehlers '25)*
× Costly determinization and boolean operations
× No efficient learning of ω-automata

The theory of deterministic parity automata is not satisfactory...

***Instead, use* Layered Automata** *but before that...*

Everybody loves automata

Let’s generalize to infinite words!

+ Finite automata ⟷ MSO-logic *(Büchi, Elgot, Trakhtenbrot ’60)*

Beautiful mathematical theory!

A minimal, canonical DFA for each regular language

Efficiency in applications!

+ Minimisation in *O(nlogn) (Hopcroft ’71)*
+ Automata learning in PTIME *(Gold ’67, Angluin 87’)*
+ Simple characterization of FO-definability

⋮

+ ω-automata ⟷ MSO-logic *(Büchi ’62)*

+ Applications in verification and reactive synthesis

No unique minimal deterministic automata

Use of ω-automata very costly in applications!

× Minimisation is NP-complete *(Schewe ’10, AlnuRadi-Ehlers ’25)*
× Costly determinization and boolean operations
× No efficient learning of ω-automata

Minimal HD coBüchi automata

Instead, use Layered Automata

but before that…

THEOREM *(Abu Radi-Kupferman '19)*

History-deterministic (HD) coBüchi automata admit minimal, canonical automata, constructible in PTIME.

THEOREM *(Abu Radi-Kupferman '19)*

Using priorities [1,2]

History-deterministic (HD) coBüchi automata admit minimal, canonical automata, constructible in PTIME.

THEOREM *(Abu Radi-Kupferman '19)*

Using priorities [1,2]

History-deterministic (HD) coBüchi automata admit minimal, canonical automata, constructible in PTIME.

An automaton is *history-deterministic* if its non-determinism can be resolved on-the-fly.

An automaton is ***history-deterministic*** if its non-determinism can be resolved on-the-fly.

= If Eve has a winning strategy in the ***letter game***:

Adam gives letters:

Eve takes transitions:

An automaton is ***history-deterministic*** if its non-determinism can be resolved on-the-fly.

= If Eve has a winning strategy in the ***letter game***:

Adam gives letters: a

Eve takes transitions:

An automaton is ***history-deterministic*** if its non-determinism can be resolved on-the-fly.

= If Eve has a winning strategy in the ***letter game***:

Adam gives letters: a

Eve takes transitions: $p_0 \xrightarrow{a} p_1$

An automaton is ***history-deterministic*** if its non-determinism can be resolved on-the-fly.

= If Eve has a winning strategy in the ***letter game***:

Adam gives letters: a b

Eve takes transitions: $p_0 \xrightarrow{a} p_1$

An automaton is ***history-deterministic*** if its non-determinism can be resolved on-the-fly.

= If Eve has a winning strategy in the ***letter game***:

Adam gives letters: a b

Eve takes transitions: $p_0 \xrightarrow{a} p_1$ $p_1 \xrightarrow{b} p_2$

An automaton is ***history-deterministic*** if its non-determinism can be resolved on-the-fly.

= If Eve has a winning strategy in the ***letter game***:

Adam gives letters: a b b a . . . $w \in \Sigma^\omega$

Eve takes transitions: $p_0 \xrightarrow{a} p_1 \quad p_1 \xrightarrow{b} p_2 \quad p_2 \xrightarrow{b} p_3$ . . . ρ run in the automaton

An automaton is ***history-deterministic*** if its non-determinism can be resolved on-the-fly.

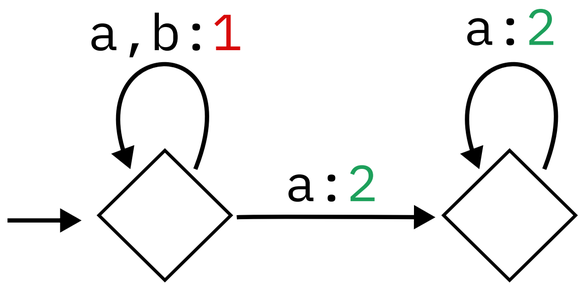
$=$ If Eve has a winning strategy in the ***letter game***:

Adam gives letters: a b b a ... $w \in \Sigma^\omega$

Eve takes transitions: $p_0 \xrightarrow{a} p_1 \quad p_1 \xrightarrow{b} p_2 \quad p_2 \xrightarrow{b} p_3 \quad \dots$ ρ run in the automaton

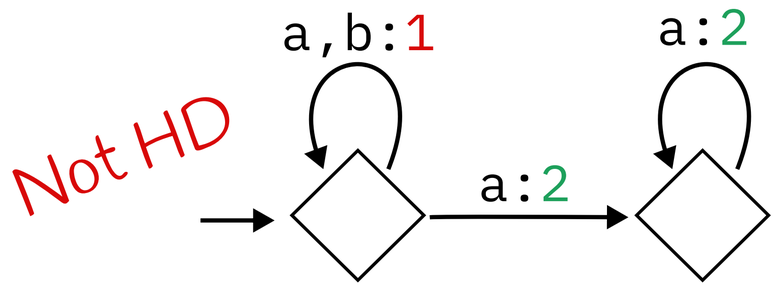
Eve wins if $w \in L(\mathcal{A}) \implies \rho$ is accepting

Examples



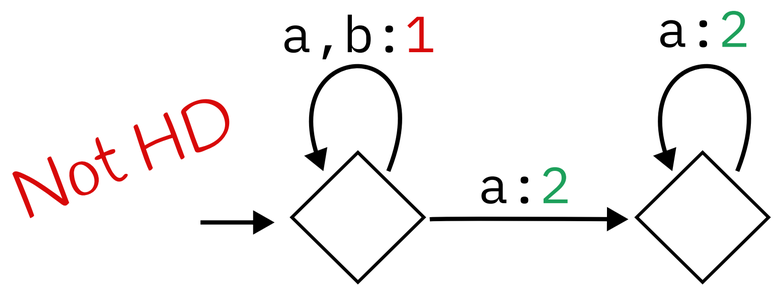
$L(A) = \texttt{Fin(b)}$

Examples

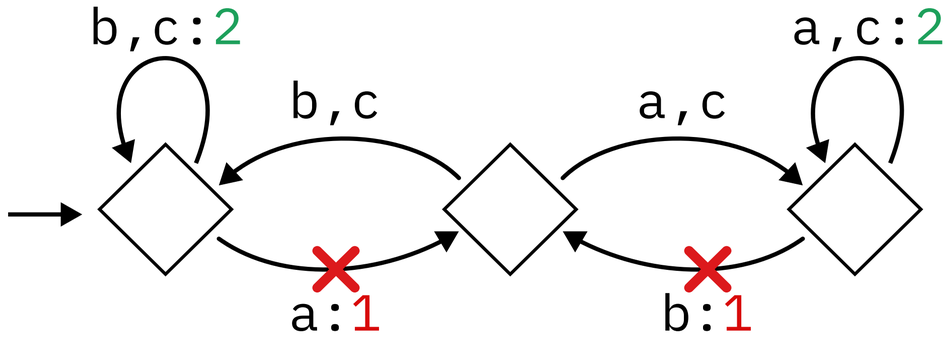


$L(A) = \texttt{Fin(b)}$

Examples

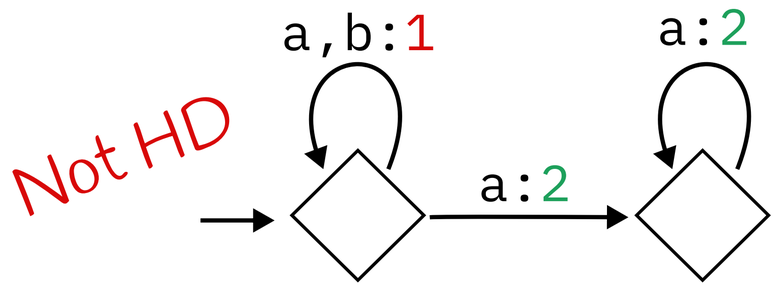


$L(A)$ = Fin(b)

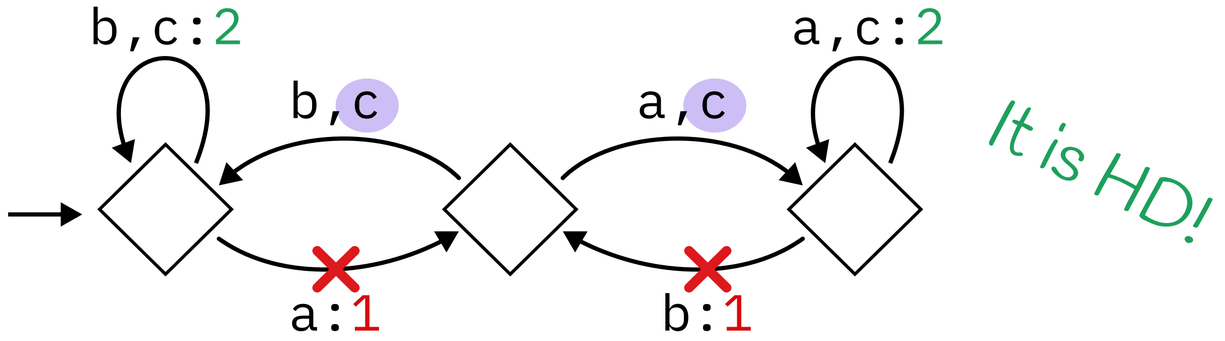


$L(A)$ = Fin(a) or Fin(b)

Examples



$L(A) = \texttt{Fin(b)}$



$L(A) = \texttt{Fin(a) or Fin(b)}$

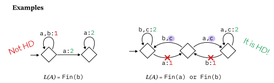
An automaton is ***history-deterministic*** if its non-determinism can be resolved on-the-fly.

$=$ If Eve has a winning strategy in the ***letter game***:

Adam gives letters: a b b a . . . $w \in \Sigma^\omega$

Eve takes transitions: $p_0 \xrightarrow{a} p_1 \quad p_1 \xrightarrow{b} p_2 \quad p_2 \xrightarrow{b} p_3$ . . . ρ run in the automaton

Eve wins if $w \in L(\mathcal{A}) \implies$ ρ is accepting



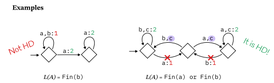
An automaton is ***history-deterministic*** if its non-determinism can be resolved on-the-fly.

= If Eve has a winning strategy in the ***letter game***:

Adam gives letters: a b b a . . . $w \in \Sigma^\omega$

Eve takes transitions: $p_0 \xrightarrow{a} p_1 \quad p_1 \xrightarrow{b} p_2 \quad p_2 \xrightarrow{b} p_3$ . . . ρ run in the automaton

Eve wins if $w \in L(\mathcal{A}) \implies$ ρ is accepting



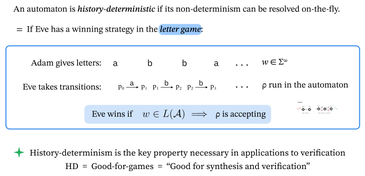
✦ History-determinism is the key property necessary in applications to verification

HD = Good-for-games = “Good for synthesis and verification”

THEOREM *(Abu Radi-Kupferman '19)*

Using priorities [1,2]

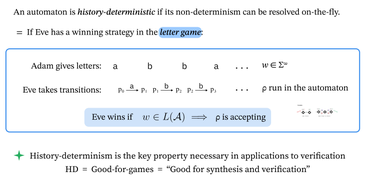
History-deterministic *(HD)* *coBüchi* automata admit minimal, canonical automata, constructible in PTIME.



THEOREM *(Abu Radi-Kupferman '19)*

Using priorities [1,2]

History-deterministic *(HD)* *coBüchi* automata admit minimal, canonical automata, constructible in PTIME.



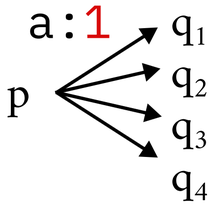
Properties of the minimal HD coBüchi automaton

Properties of the minimal HD coBüchi automaton

A. If $p \xrightarrow{a:2} q$, then this is the only a-transition from p (*2-deterministic*)

Properties of the minimal HD coBüchi automaton

A. If $p \xrightarrow{a:2} q$, then this is the only a-transition from p (*2-deterministic*)

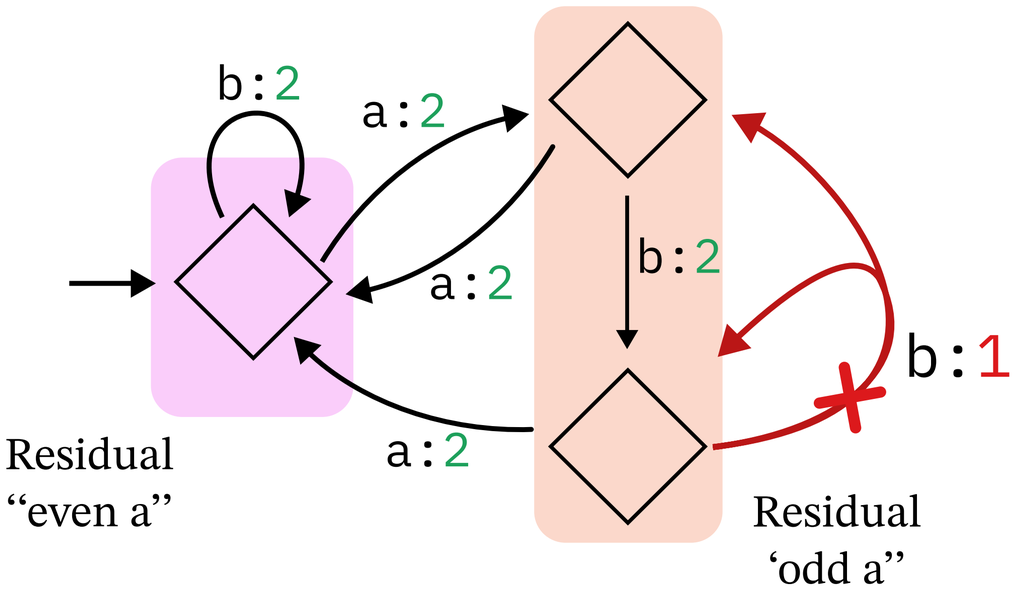
B. If $p \xrightarrow{a:1} q$ (with a crossed-out arrow), then also 

to all states that are languages-equivalent to q *(1-saturated)*

Properties of the minimal HD coBüchi automaton

A. If $p \xrightarrow{a:2} q$, then this is the only a-transition from p (*2-deterministic*)

B. If $p \xrightarrow{a:1} q$, then also $p \xrightarrow{a:1} q_1, q_2, q_3, q_4$ to all states that are languages-equivalent to q *(1-saturated)*



An automaton is *history-deterministic* if its non-determinism can be resolved on-the-fly.

= If Eve has a winning strategy in the *letter game*:

Adam gives letters: a b b a ... $w \in \Sigma^\omega$

Eve takes transitions: $p_0 \xrightarrow{a} p_1 \quad p_1 \xrightarrow{b} p_2 \quad p_2 \xrightarrow{b} p_3 \quad \dots$ ρ run in the automaton

Eve wins if $w \in L(\mathcal{A}) \implies \rho$ is accepting

+ History-determinism is the key property necessary in applications to verification

HD = Good-for-games = "Good for synthesis and verification"

THEOREM *(Abu Radi-Kupferman '19)*

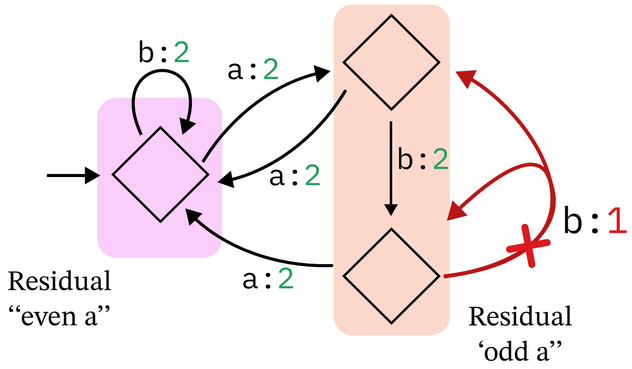
Using priorities [1,2]

History-deterministic *(HD)* *coBüchi* automata admit minimal, canonical automata, constructible in PTIME.

Properties of the minimal HD coBüchi automaton

A. If $p \xrightarrow{a:2} q$, then this is the only a-transition from p *(2-deterministic)*

B. If $p \xrightarrow{a:1} \times q$, then also $p \xrightarrow{a:1} q_1, q_2, q_3, q_4$ to all states that are languages-equivalent to q *(1-saturated)*



THEOREM *(Folklore)*

CoBüchi automata with properties A. and B. are ***0-1-probabilistic***.

THEOREM *(Folklore)*

CoBüchi automata with properties A. and B. are ***0-1-probabilistic***.

$\mathcal{A}$ is ***0-1-probabilistic*** if for every $w \in L(\mathcal{A})$

a random run over w is accepting with probability 1.

THEOREM *(Folklore)*

CoBüchi automata with properties A. and B. are ***0-1-probabilistic***.

$\mathcal{A}$ is ***0-1-probabilistic*** if for every $w \in L(\mathcal{A})$
a random run over w is accepting with probability 1.

Useful in verification of probabilistic systems!

THEOREM *(Folklore)*

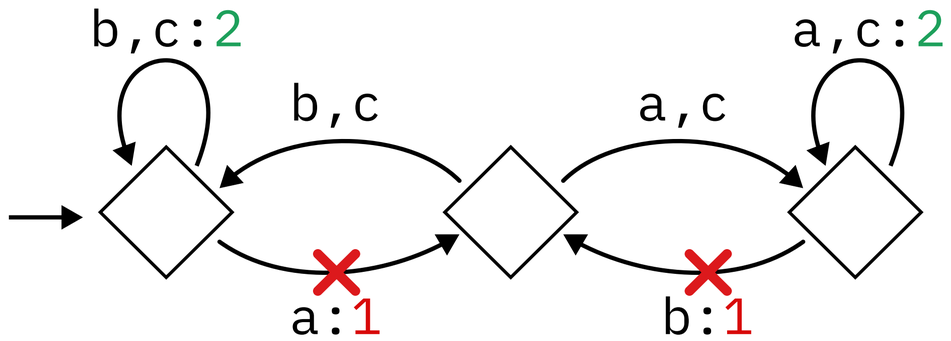
CoBüchi automata with properties A. and B. are ***0-1-probabilistic***.

$\mathcal{A}$ is ***0-1-probabilistic*** if for every $w \in L(\mathcal{A})$

a random run over w is accepting with probability 1.

Useful in verification of probabilistic systems!

Example:



$L(A)$ = `Fin(a) or Fin(b)`

An automaton is ***history-deterministic*** if its non-determinism can be resolved on-the-fly.

= If Eve has a winning strategy in the *letter game*:

Adam gives letters: a b b a ... $w \in \Sigma^\omega$

Eve takes transitions: $p_0 \xrightarrow{a} p_1 \quad p_1 \xrightarrow{b} p_2 \quad p_2 \xrightarrow{b} p_1 \quad \dots$ ρ run in the automaton

Eve wins if $w \in L(\mathcal{A}) \implies \rho$ is accepting

+ History-determinism is the key property necessary in applications to verification

HD = Good-for-games = "Good for synthesis and verification"

THEOREM *(Abu Radi-Kupferman '19)*

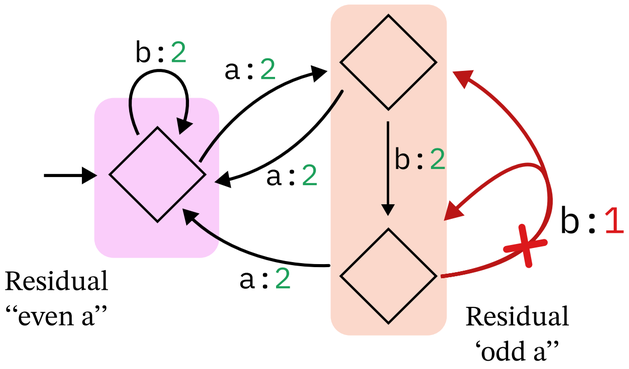
Using priorities [1,2]

History-deterministic *(HD)* *coBüchi* automata admit minimal, canonical automata, constructible in PTIME.

Properties of the minimal HD coBüchi automaton

A. If $p \xrightarrow{a:2} q$, then this is the only a-transition from p (*2-deterministic*)

B. If $p \xrightarrow{a:1} q$, then also $p \xrightarrow{a:1} q_1, q_2, q_3, q_4$ to all states that are languages-equivalent to q *(1-saturated)*



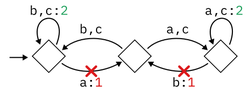
THEOREM *(Folklore)*

CoBüchi automata with properties A. and B. are ***0-1-probabilistic***.

$\mathcal{A}$ is ***0-1-probabilistic*** if for every $w \in L(\mathcal{A})$ a random run over w is accepting with probability 1.

Useful in verification of probabilistic systems!

Example:



$L(A)$ = Fin(a) or Fin(b)

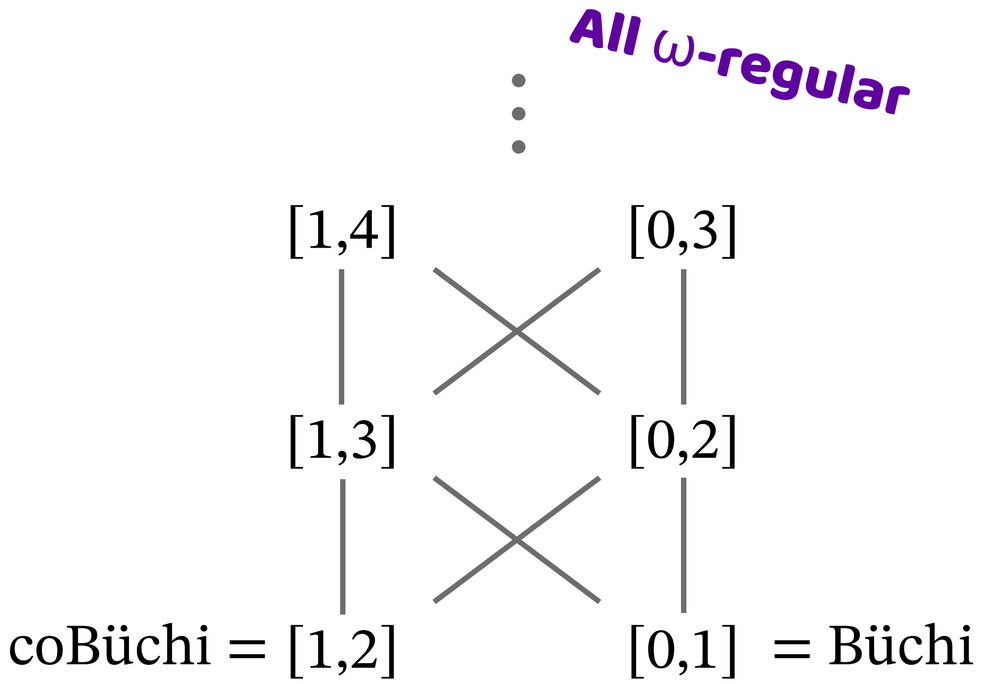
Minimal HD coBüchi are great!

Minimal HD coBüchi are great!

But they only recognize a small part of all ω-regular languages

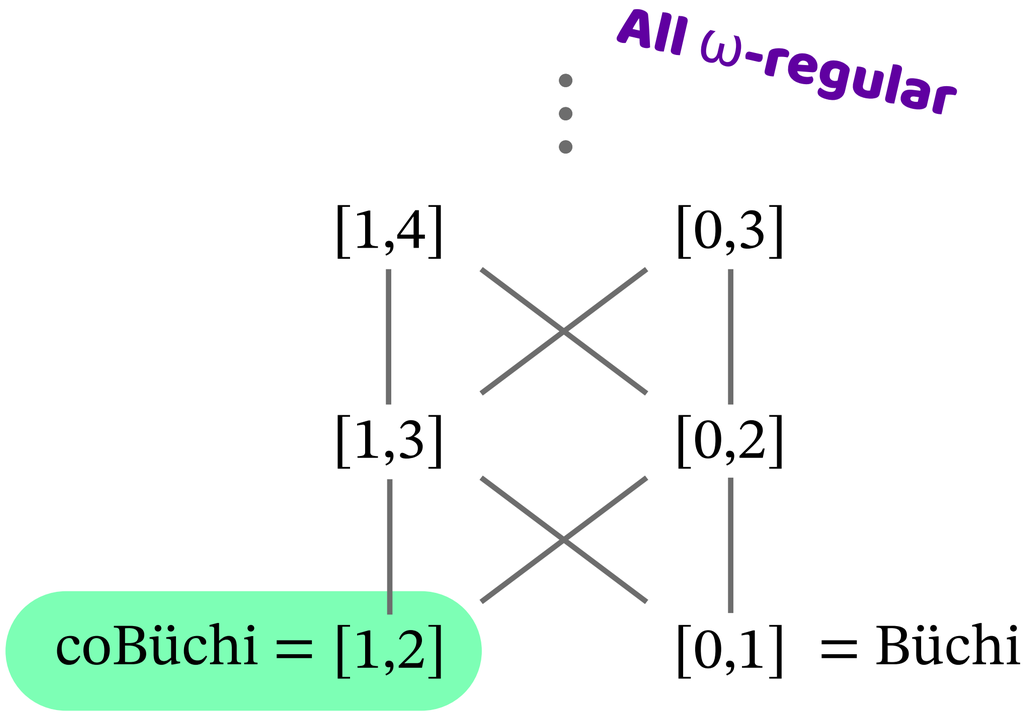
Minimal HD coBüchi are great!

But they only recognize a small part of all ω-regular languages



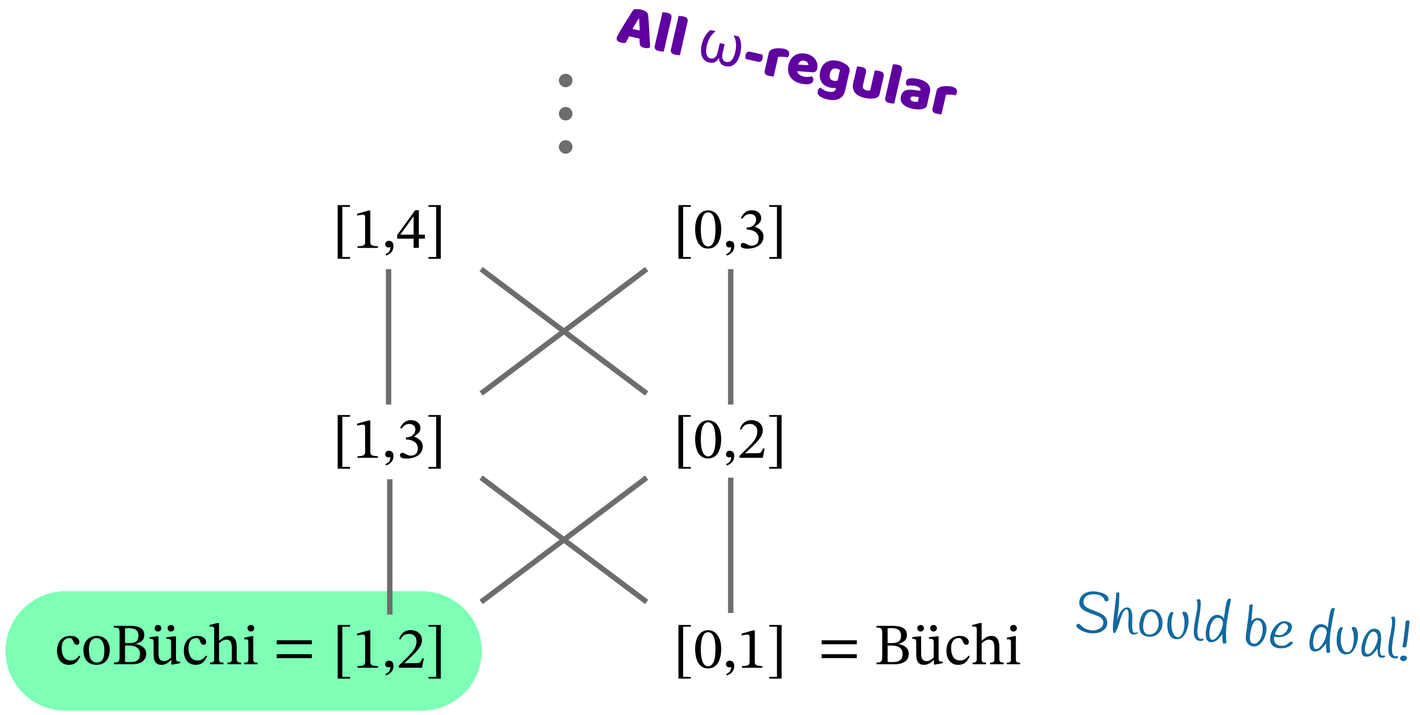
Minimal HD coBüchi are great!

But they only recognize a small part of all ω-regular languages



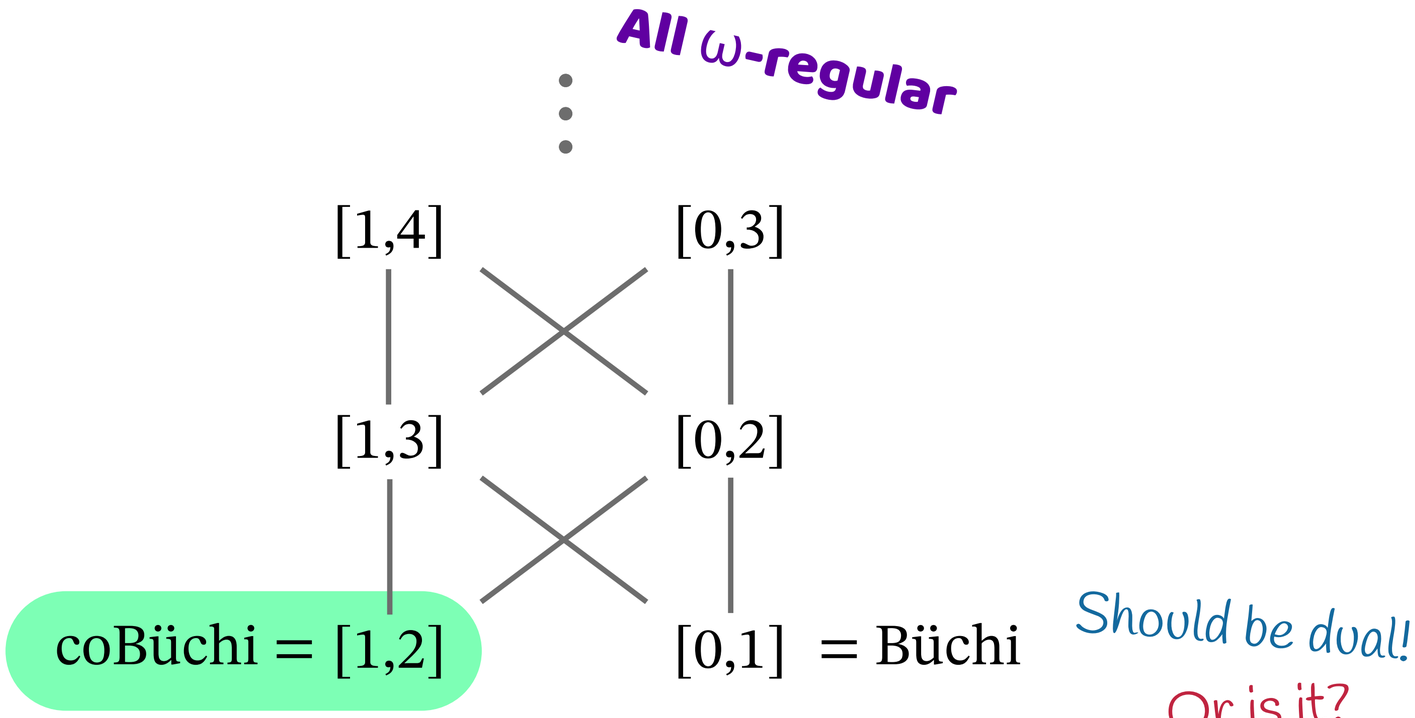
Minimal HD coBüchi are great!

But they only recognize a small part of all ω-regular languages



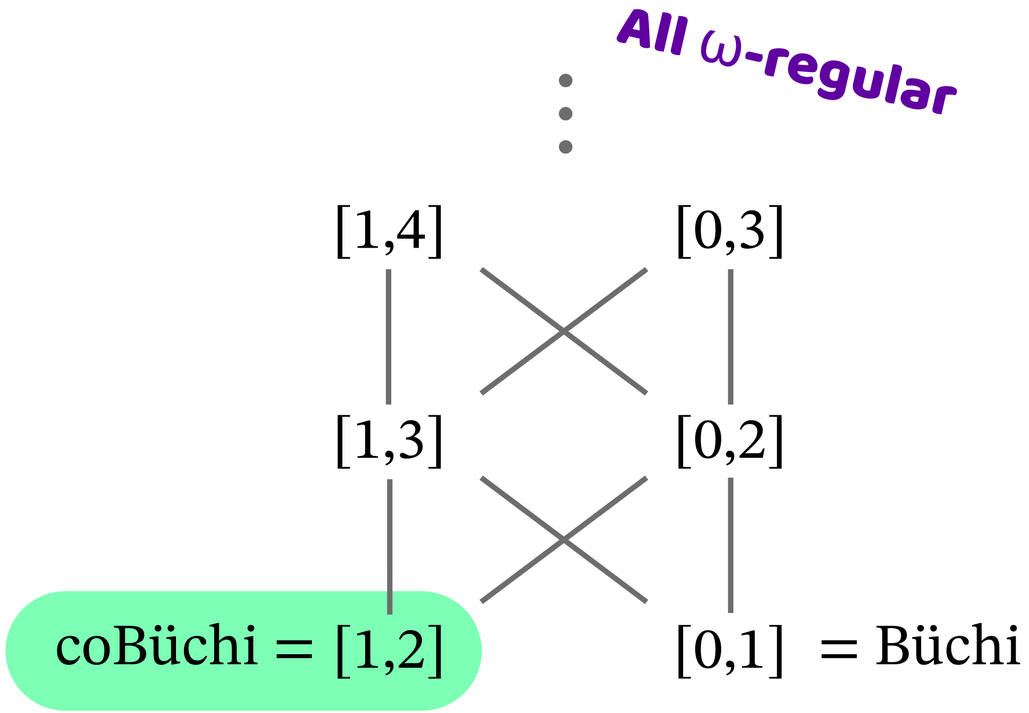
Minimal HD coBüchi are great!

But they only recognize a small part of all ω-regular languages



Minimal HD coBüchi are great!

But they only recognize a small part of all ω-regular languages



Büchi

THEOREM *(Abu Radi-Ehlers '25)*

The minimisation of HD Büchi automata is NP-complete.

Büchi

THEOREM *(Abu Radi-Ehlers '25)*

The minimisation of HD Büchi automata is NP-complete.

THEOREM *(Corollary of coBüchi case)*

HD ***universal*** Büchi automata can be minimised in polynomial time.

Büchi

THEOREM *(Abu Radi-Ehlers '25)*

The minimisation of HD Büchi automata is NP-complete.

THEOREM *(Corollary of coBüchi case)*

HD ***universal*** Büchi automata can be minimised in polynomial time.

HD *universal* Büchi automata can

- Non-Det automata: w is accepted if ***some*** run is accepting.

HD *universal* Büchi automata can

- Non-Det automata: w is accepted if ***some*** run is accepting.

  An existential player tries to build an accepting run.

HD *universal* Büchi automata can

- Non-Det automata: w is accepted if ***some*** run is accepting.

  An existential player tries to build an accepting run.

- Universal automata: w is accepted if ***all*** run are accepting

HD *universal* Büchi automata can

- Non-Det automata: w is accepted if ***some*** run is accepting.

  An existential player tries to build an accepting run.

- Universal automata: w is accepted if ***all*** run are accepting
  w is rejected if ***some*** run is rejecting.

HD *universal* Büchi automata can

- Non-Det automata: w is accepted if ***some*** run is accepting.

  An existential player tries to build an accepting run.

- Universal automata: w is accepted if ***all*** run are accepting
  w is rejected if ***some*** run is rejecting.

  A universal player tries to build a rejecting run.

HD ***universal*** Büchi automata can

- Non-Det automata: w is accepted if ***some*** run is accepting.

  An existential player tries to build an accepting run.

- Universal automata: w is accepted if ***all*** run are accepting
  w is rejected if ***some*** run is rejecting.

  A universal player tries to build a rejecting run.

Automata are ***history-deterministic*** if the respective player has a strategy to do this on-the-fly

Büchi

THEOREM *(Abu Radi-Ehlers '25)*

The minimisation of HD Büchi automata is NP-complete.

THEOREM *(Corollary of coBüchi case)*

HD ***universal*** Büchi automata can be minimised in polynomial time.

- Non-Det automata: w is accepted if ***some*** run is accepting.

  An existential player tries to build an accepting run.

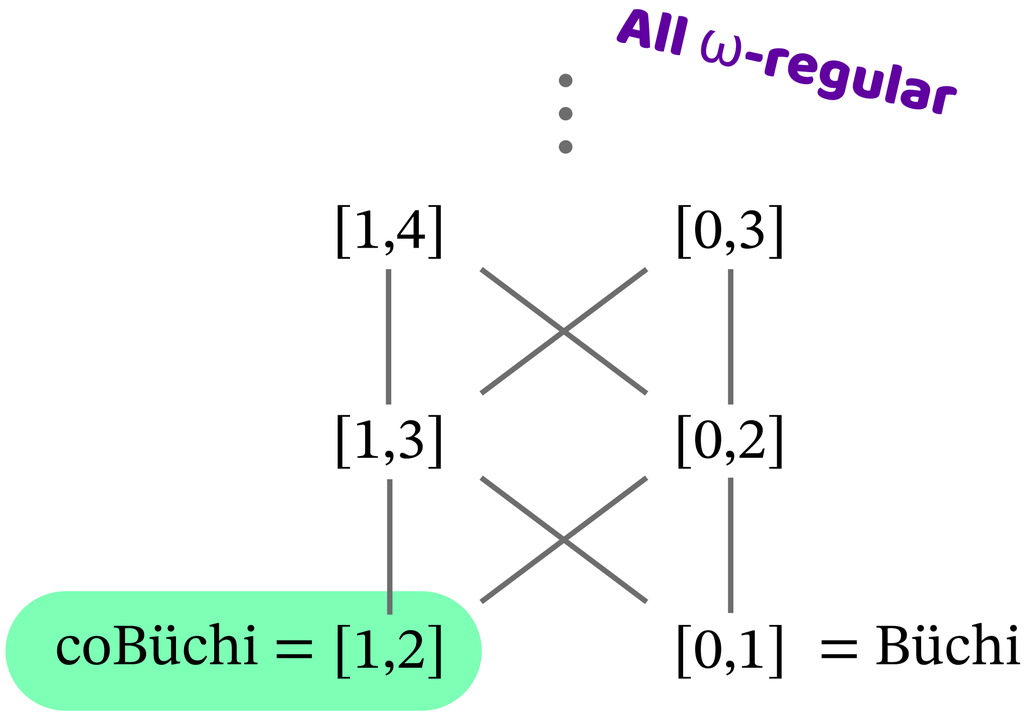
- Universal automata: w is accepted if ***all*** run are accepting
  w is rejected if ***some*** run is rejecting.

  A universal player tries to build a rejecting run.

Automata are ***history-deterministic*** if the respective player has a strategy to do this on-the-fly

Minimal HD coBüchi are great!

But they only recognize a small part of all ω-regular languages



THEOREM *(Abu Radi-Ehlers '25)*

The minimisation of HD Büchi automata is NP-complete.

THEOREM *(Corollary of coBüchi case)*

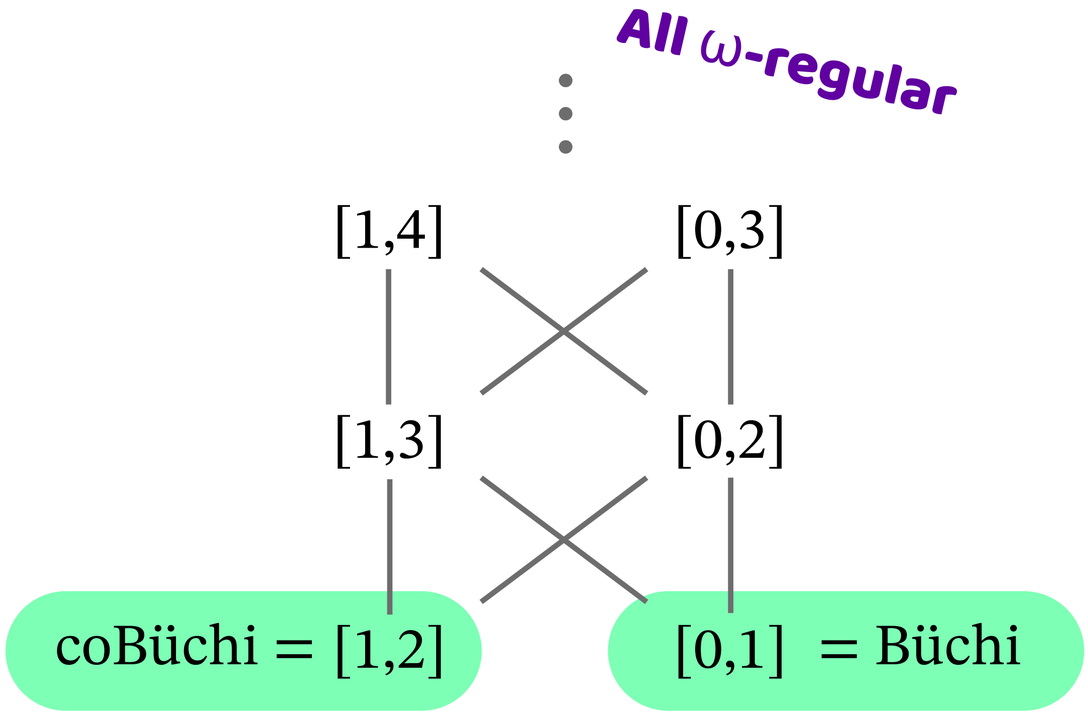
HD ***universal*** Büchi automata can be minimised in polynomial time.

- Non-Det automata: *w* is accepted if ***some*** run is accepting.
  An existential player tries to build an accepting run.
- Universal automata: *w* is accepted if ***all*** runs are accepting
  or is rejected if ***some*** run is rejecting.
  A universal player tries to build a rejecting run.

Automata are **history-deterministic** if the respective player has a strategy to do this on-the-fly

Minimal HD coBüchi are great!

But they only recognize a small part of all ω-regular languages



THEOREM *(Abu Radi-Ehlers '25)*

The minimisation of HD Büchi automata is NP-complete.

THEOREM *(Corollary of coBüchi case)*

HD ***universal*** Büchi automata can be minimised in polynomial time.

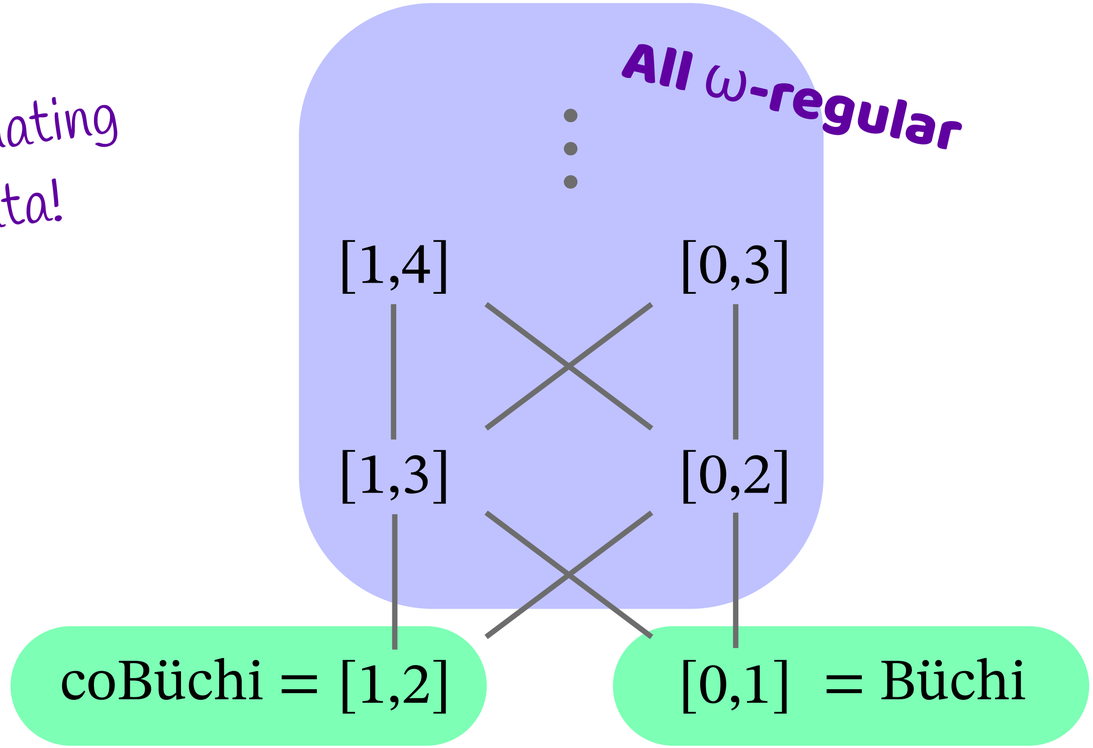
- Non-Det automata: w is accepted if ***some*** run is accepting.
  An existential player tries to build an accepting run.
- Universal automata: w is accepted if ***all*** runs are accepting
  w is rejected if ***some*** run is rejecting.
  A universal player tries to build a rejecting run.

Automata are **history-deterministic** if the respective player has a strategy to do this on-the-fly

Minimal HD coBüchi are great!

But they only recognize a small part of all ω-regular languages

Use alternating automata!



THEOREM *(Abu Radi-Ehlers '25)*

The minimisation of HD Büchi automata is NP-complete.

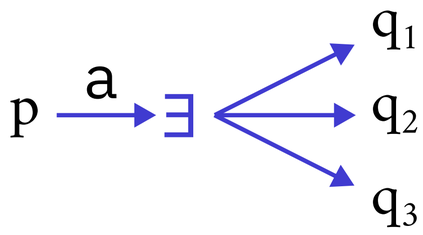
THEOREM *(Corollary of coBüchi case)*

HD ***universal*** Büchi automata can be minimised in polynomial time.

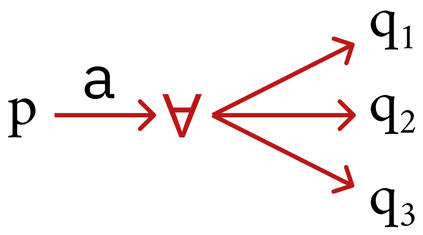
- Non-Det automata: w is accepted if ***some*** run is accepting.
  An existential player tries to build an accepting run.
- Universal automata: w is accepted if ***all*** run are accepting
  or is rejected if ***some*** run is rejecting.
  A universal player tries to build a rejecting run.

Automata are **history-deterministic** if the respective player has a strategy to do this on-the-fly

Alternating automata

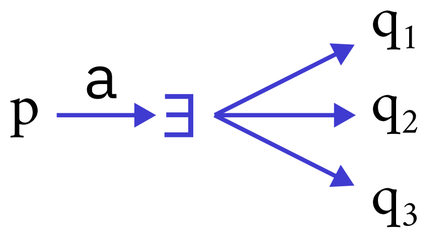


existential choices

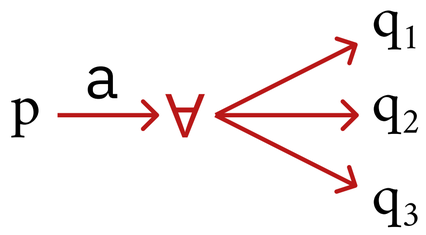


universal choices

Alternating automata



existential choices

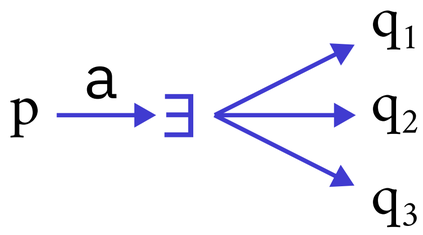


universal choices

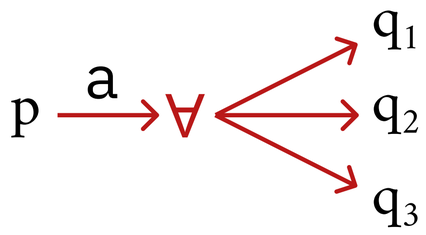
A word w induces a parity game.

$w \in L(\mathcal{A})$ if the existential player wins this game.

Alternating automata



existential choices



universal choices

A word w induces a parity game.

$w \in L(\mathcal{A})$ if the existential player wins this game.

An alternating automaton is ***history-deterministic*** if both players have on-the-fly strategies in these games.

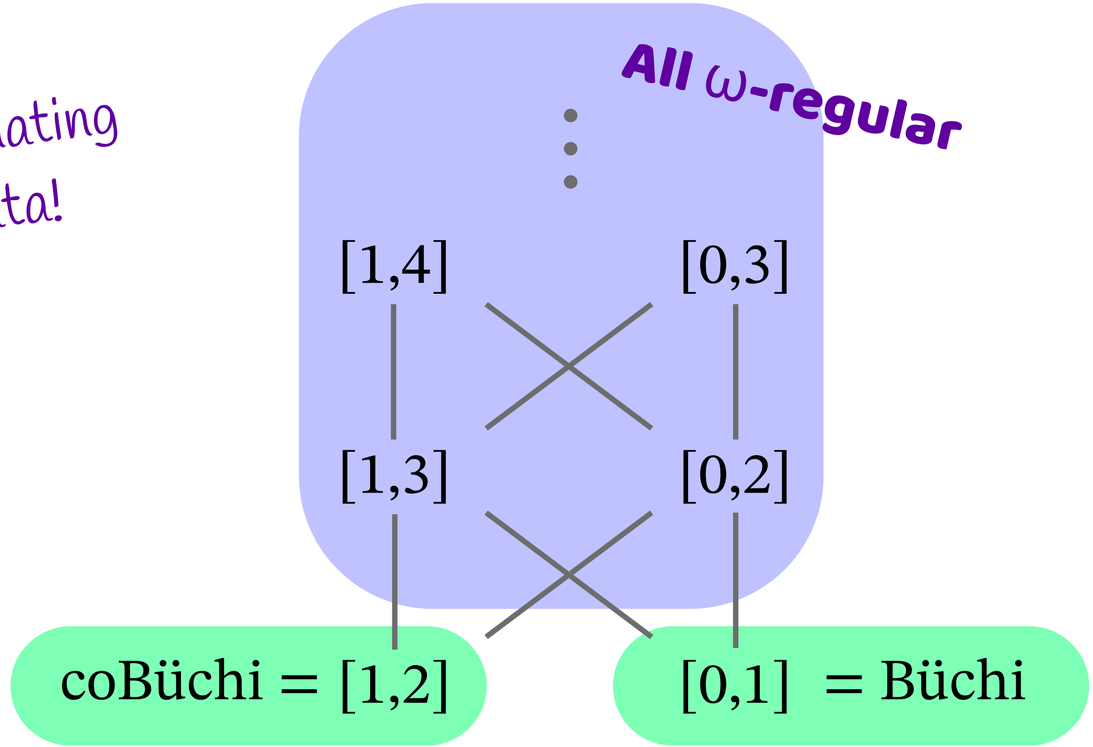
THEOREM *(C'-Löding-Walukiewicz)*

The minimal HD non-deterministic coBüchi automaton is also minimal amongst all HD alternating (co)Büchi automata.

Minimal HD coBüchi are great!

But they only recognize a small part of all ω-regular languages

Use alternating automata!



Alternating automata

existential choices universal choices

A word w induces a parity game.
$w \in L(\mathcal{A})$ if the existential player wins this game.

An alternating automaton is ***history-deterministic*** if both players have on-the-fly strategies in these games.

THEOREM *(C*-Löding-Walukiewicz)*

The minimal HD non-deterministic coBüchi automaton is also minimal amongst all HD alternating (co)Büchi automata.

THEOREM *(Abu Radi-Ehlers '25)*

The minimisation of HD Büchi automata is NP-complete.

THEOREM *(Corollary of coBüchi case)*

HD ***universal*** Büchi automata can be minimised in polynomial time.

- Non-Det automata: w is accepted if ***some*** run is accepting.
  An existential player tries to build an accepting run.
- Universal automata: w is accepted if ***all*** run are accepting
  w is rejected if ***some*** run is rejecting.
  A universal player tries to build a rejecting run.

Automata are ***history-deterministic*** if the respective player has a strategy to do this on-the-fly

Everybody loves automata

- Finite automata ⟷ MSO-logic *(Büchi, Elgot, Trakhtenbrot '60)*

Beautiful mathematical theory!

A minimal, canonical DFA for each regular language

Efficiency in applications!

- Minimisation in $O(n\log n)$ *(Hopcroft '71)*
- Automata learning in PTIME *(Gold '67, Angluin 87')*
- Simple characterization of FO-definability

⋮

Let's generalize to infinite words!

- ω-automata ⟷ MSO-logic *(Büchi '62)*
- Applications in verification and reactive synthesis

No unique minimal deterministic automata

Use of ω-automata very costly in applications!

- Minimisation is NP-complete *(Schewe '10, AbuRadi-Ehlers '25)*
- Costly determinization and boolean operations
- No efficient learning of ω-automata

Minimal HD coBüchi automata

Instead, use Layered Automata *but before that...*

THEOREM *(Abu Radi-Kupferman '19)*

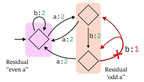
Using priorities {1,2}

History-deterministic (HD) coBüchi automata admit minimal, canonical automata, constructible in PTIME.

Properties of the minimal HD coBüchi automaton

A. If $p \xrightarrow{a:2} q$, then this is the only a-transition from p *(2-deterministic)*

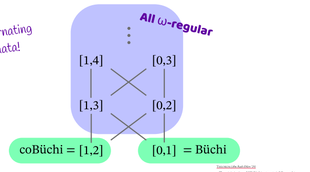
B. If $p \not\xrightarrow{a:2} q$, then also $p \xrightarrow{a:\{1,2\}} \{q_1, q_2, q_3, q_4\}$ to all states that are languages-equivalent to q *(1-saturated)*



Minimal HD coBüchi are great!

But they only recognize a small part of all ω-regular languages

Use alternating automata!



Everybody loves automata

- Finite automata ⟷ MSO-logic *(Büchi, Elgot, Trakhtenbrot '60)*

Beautiful mathematical theory!

A minimal, canonical DFA for each regular language

Efficiency in applications!

- Minimisation in $O(n\log n)$ *(Hopcroft '71)*
- Automata learning in PTIME *(Gold '67, Angluin 87')*
- Simple characterization of FO-definability

⋮

Let's generalize to infinite words!

- ω-automata ⟷ MSO-logic *(Büchi '62)*
- Applications in verification and reactive synthesis

No unique minimal deterministic automata

Use of ω-automata very costly in applications!

- Minimisation is NP-complete *(Schewe '10, AbuRadi-Ehlers '25)*
- Costly determinization and boolean operations
- No efficient learning of ω-automata

Minimal HD coBüchi automata

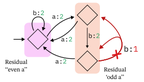
THEOREM *(Abu Radi-Kupferman '19)* — Using priorities {1,2}

History-deterministic (HD) *coBüchi* automata admit minimal, canonical automata, constructible in PTIME.

Properties of the minimal HD coBüchi automaton

A. If $p \xrightarrow{a:2} q$, then this is the only a-transition from p *(2-deterministic)*

B. If $p \not\xrightarrow{a:2} q$, then also $p \xrightarrow{a:1} q_0, q_1, q_2, q_3$ to all states that are languages-equivalent to q *(1-saturated)*



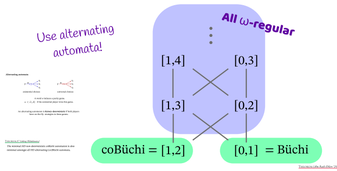
Residual "even a"

Residual "odd a"

Minimal HD coBüchi are great!

But they only recognize a small part of all ω-regular languages

Use alternating automata!



Instead, use ✨ ***Layered Automata*** ✨

Layered Automata

Formalism to represent a subclass of alternating parity automata

Layered Automata

Formalism to represent a subclass of alternating parity automata

Definition

Layered automaton: A sequence of deterministic transition systems with morphisms

$$\mathcal{T}_1 \xleftarrow{\mu_1} \mathcal{T}_2 \xleftarrow{\mu_2} \quad \dots \quad \mathcal{T}_{d-1} \xleftarrow{\mu_{d-1}} \mathcal{T}_d$$

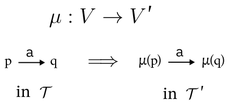
with morphisms

$$\mu : V \to V'$$

$$\mathrm{p} \xrightarrow{\mathtt{a}} \mathrm{q} \quad \text{in } \mathcal{T} \qquad \Longrightarrow \qquad \mu(\mathrm{p}) \xrightarrow{\mathtt{a}} \mu(\mathrm{q}) \quad \text{in } \mathcal{T}'$$

Layered automaton: A sequence of deterministic transition systems with morphisms

$$\mathcal{T}_1 \xleftarrow{\mu_1} \mathcal{T}_2 \xleftarrow{\mu_2} \quad \dots \quad \mathcal{T}_{d-1} \xleftarrow{\mu_{d-1}} \mathcal{T}_d$$



incomplete

Layered automaton: A sequence of deterministic transition systems with morphisms

$$\mathcal{T}_1 \xleftarrow{\mu_1} \mathcal{T}_2 \xleftarrow{\mu_2} \quad \dots \quad \mathcal{T}_{d-1} \xleftarrow{\mu_{d-1}} \mathcal{T}_d$$

$\mu : V \to V'$

$p \xrightarrow{a} q$ in $\mathcal{T}$ $\implies$ $\mu(p) \xrightarrow{a} \mu(q)$ in $\mathcal{T}'$

incomplete

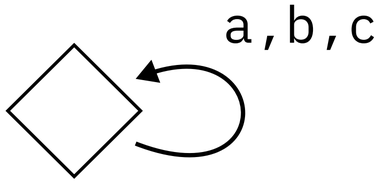
Layered automaton: A sequence of deterministic transition systems with morphisms

$$\mathcal{T}_1 \xleftarrow{\mu_1} \mathcal{T}_2 \xleftarrow{\mu_2} \quad \dots \quad \mathcal{T}_{d-1} \xleftarrow{\mu_{d-1}} \mathcal{T}_d$$

$\mu : V \to V'$

$p \xrightarrow{a} q$ in $\mathcal{T}$ $\implies$ $\mu(p) \xrightarrow{a} \mu(q)$ in $\mathcal{T}'$

$\mathcal{T}_1$



incomplete

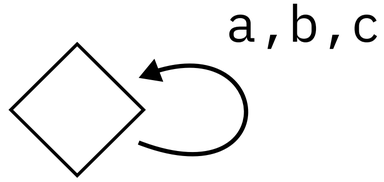
Layered automaton: A sequence of deterministic transition systems with morphisms

$$\mathcal{T}_1 \xleftarrow{\mu_1} \mathcal{T}_2 \xleftarrow{\mu_2} \quad \dots \quad \mathcal{T}_{d-1} \xleftarrow{\mu_{d-1}} \mathcal{T}_d$$

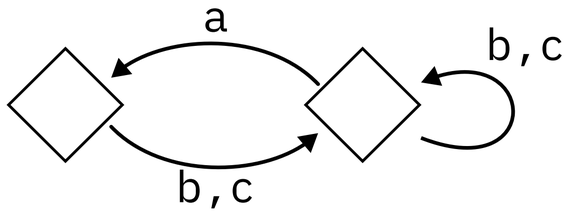
$\mu : V \to V'$

$p \xrightarrow{a} q$ in $\mathcal{T}$ $\implies$ $\mu(p) \xrightarrow{a} \mu(q)$ in $\mathcal{T}'$

$\mathcal{T}_1$



$\mathcal{T}_2$



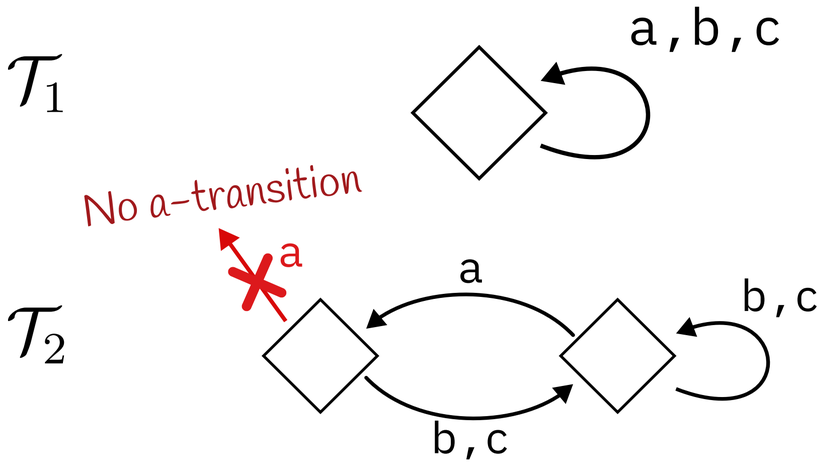
incomplete

Layered automaton: A sequence of deterministic transition systems with morphisms

$$\mathcal{T}_1 \xleftarrow{\mu_1} \mathcal{T}_2 \xleftarrow{\mu_2} \quad \dots \quad \mathcal{T}_{d-1} \xleftarrow{\mu_{d-1}} \mathcal{T}_d$$

$\mu : V \to V'$

$p \xrightarrow{a} q$ in $\mathcal{T}$ $\implies$ $\mu(p) \xrightarrow{a} \mu(q)$ in $\mathcal{T}'$



incomplete

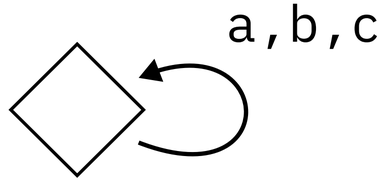
Layered automaton: A sequence of deterministic transition systems with morphisms

$$\mathcal{T}_1 \xleftarrow{\mu_1} \mathcal{T}_2 \xleftarrow{\mu_2} \quad \dots \quad \mathcal{T}_{d-1} \xleftarrow{\mu_{d-1}} \mathcal{T}_d$$

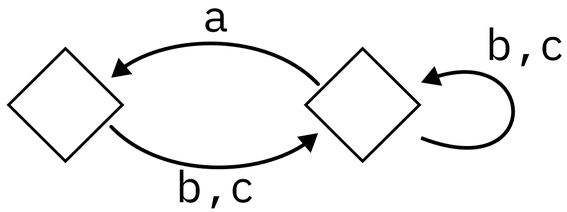
$\mu : V \to V'$

$p \xrightarrow{a} q$ in $\mathcal{T}$ $\implies$ $\mu(p) \xrightarrow{a} \mu(q)$ in $\mathcal{T}'$

$\mathcal{T}_1$



$\mathcal{T}_2$



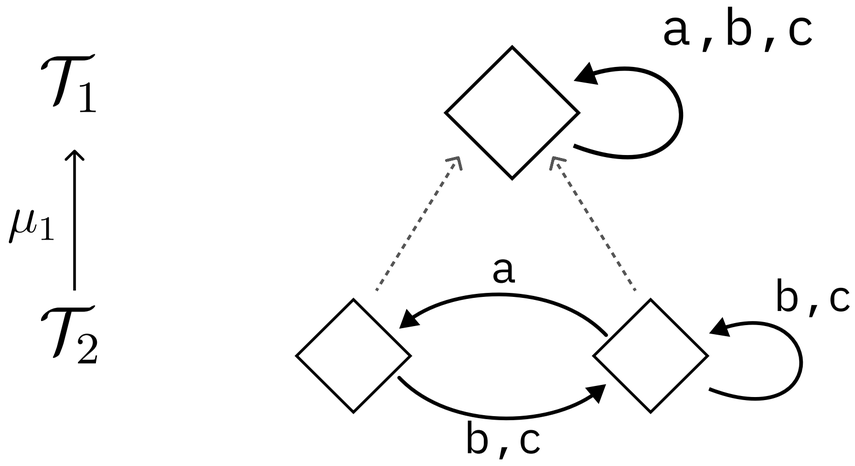
incomplete

Layered automaton: A sequence of deterministic transition systems with morphisms

$$\mathcal{T}_1 \xleftarrow{\mu_1} \mathcal{T}_2 \xleftarrow{\mu_2} \quad \dots \quad \mathcal{T}_{d-1} \xleftarrow{\mu_{d-1}} \mathcal{T}_d$$

$\mu : V \to V'$

$p \xrightarrow{a} q$ in $\mathcal{T}$ $\implies$ $\mu(p) \xrightarrow{a} \mu(q)$ in $\mathcal{T}'$



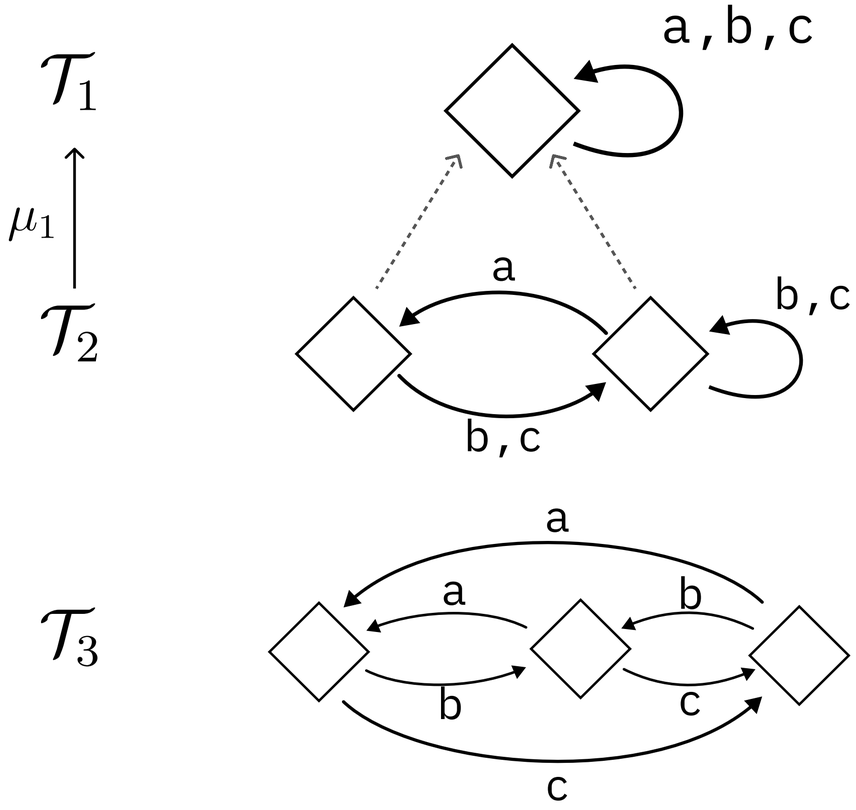
incomplete

Layered automaton: A sequence of deterministic transition systems with morphisms

$$\mathcal{T}_1 \xleftarrow{\mu_1} \mathcal{T}_2 \xleftarrow{\mu_2} \quad \dots \quad \mathcal{T}_{d-1} \xleftarrow{\mu_{d-1}} \mathcal{T}_d$$

$\mu : V \to V'$

$p \xrightarrow{a} q$ in $\mathcal{T}$ $\implies$ $\mu(p) \xrightarrow{a} \mu(q)$ in $\mathcal{T}'$



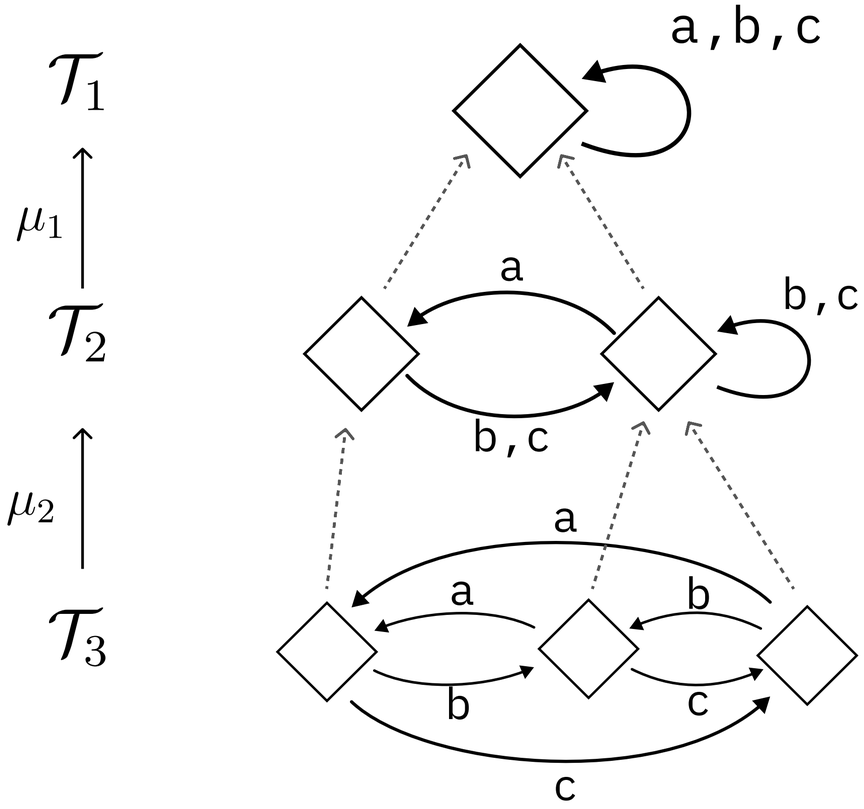
incomplete

Layered automaton: A sequence of deterministic transition systems with morphisms

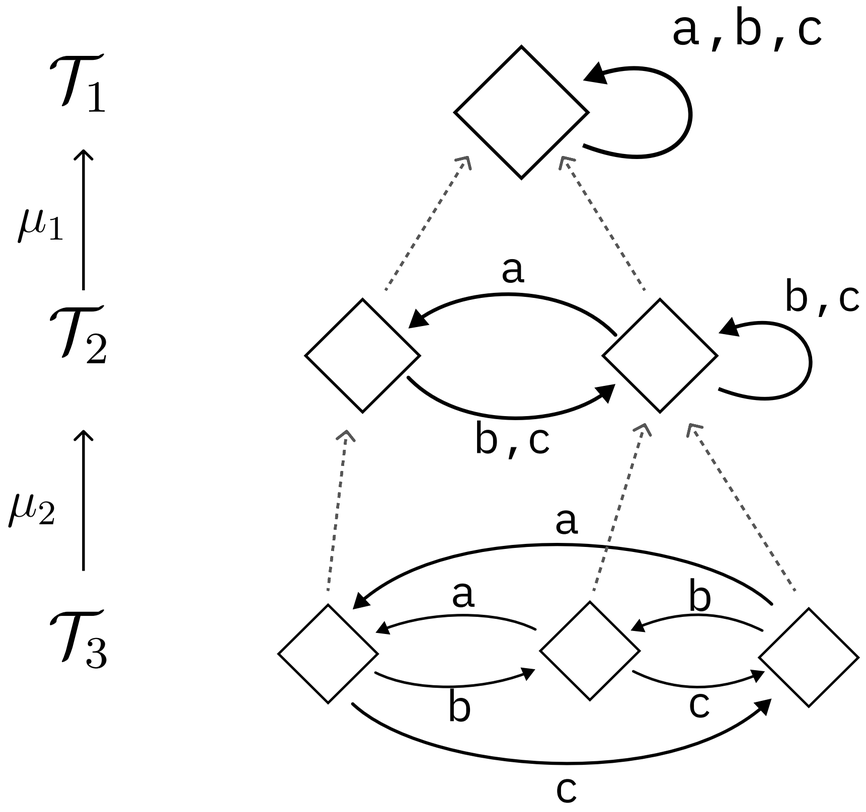
$$\mathcal{T}_1 \xleftarrow{\mu_1} \mathcal{T}_2 \xleftarrow{\mu_2} \quad \dots \quad \mathcal{T}_{d-1} \xleftarrow{\mu_{d-1}} \mathcal{T}_d$$

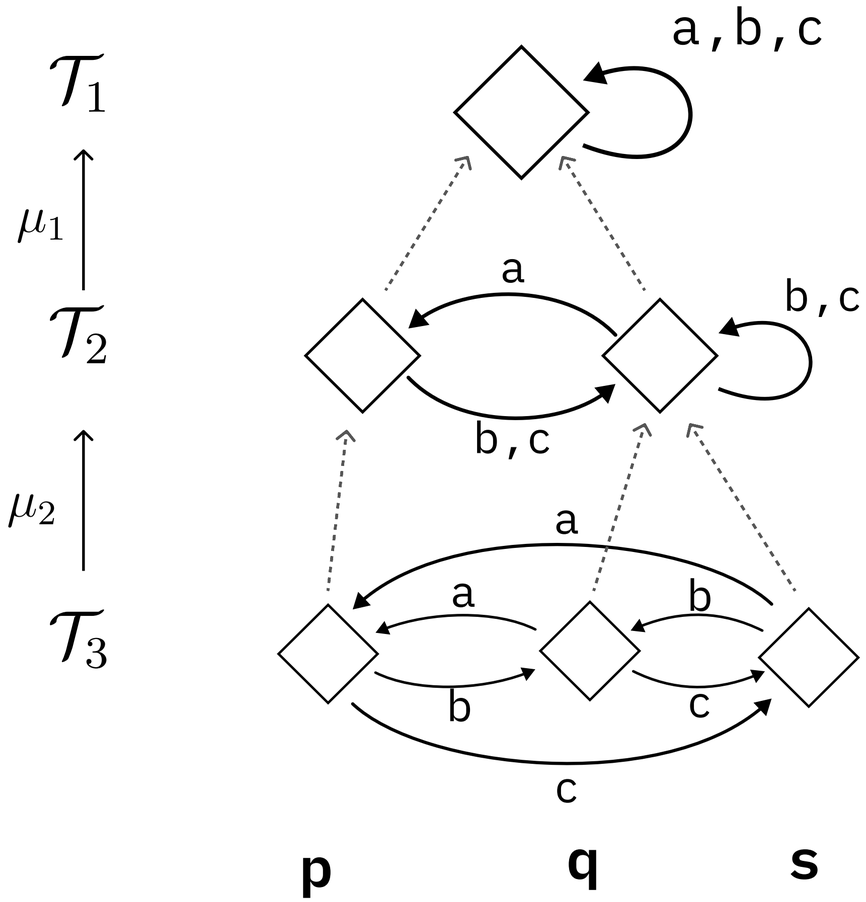
$\mu : V \to V'$

$p \xrightarrow{a} q \implies \mu(p) \xrightarrow{a} \mu(q)$ in $\mathcal{T}$ / in $\mathcal{T}'$



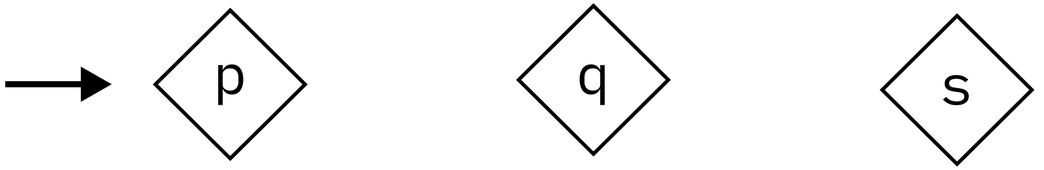
Alternating Parity Automaton

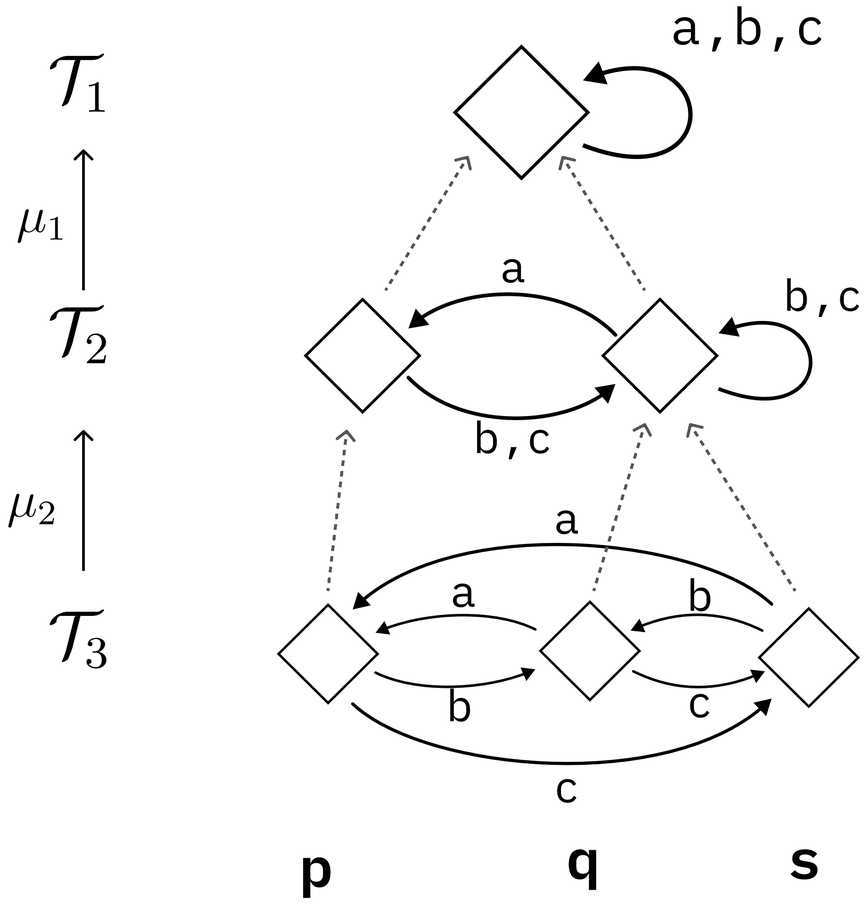




Alternating Parity Automaton

States: leaves

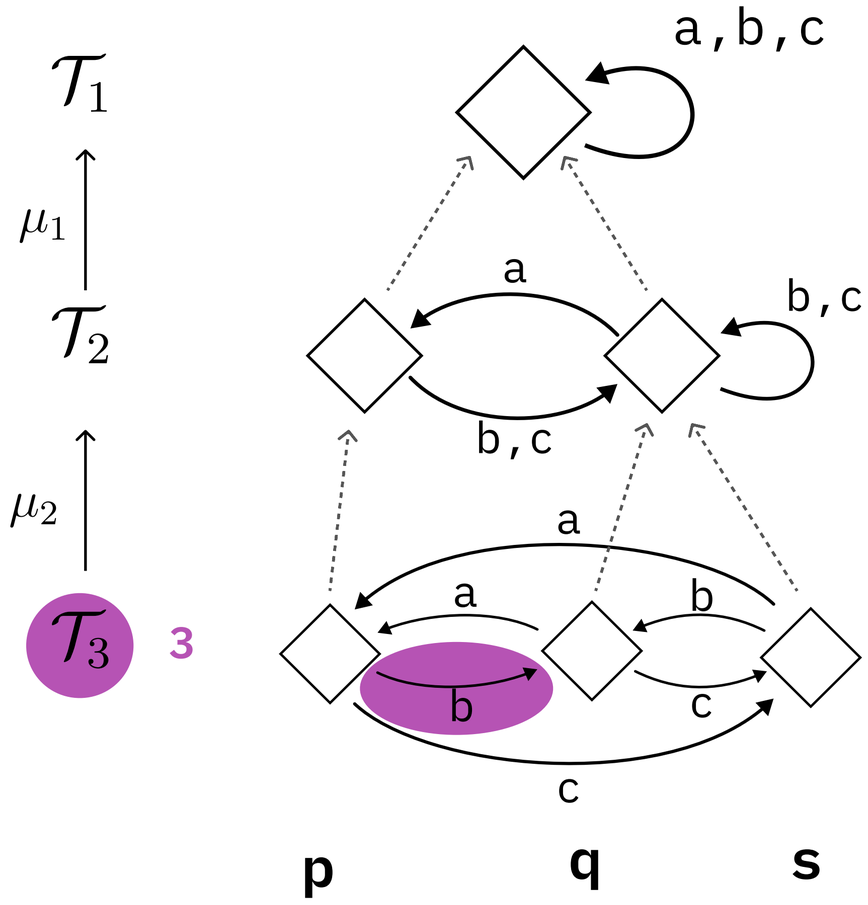




Alternating Parity Automaton

States: leaves

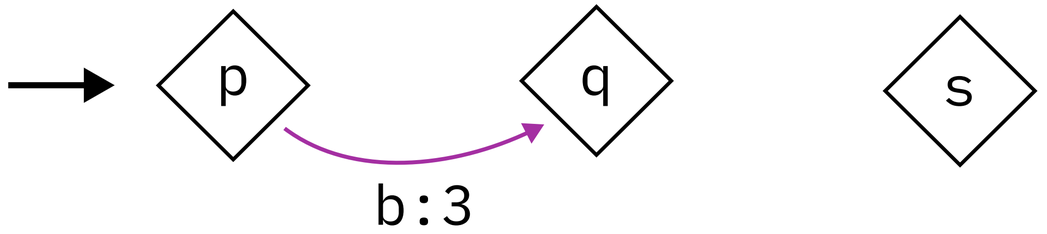
Transitions: given by the deepest $\mathcal{T}_i$ where the transition exists

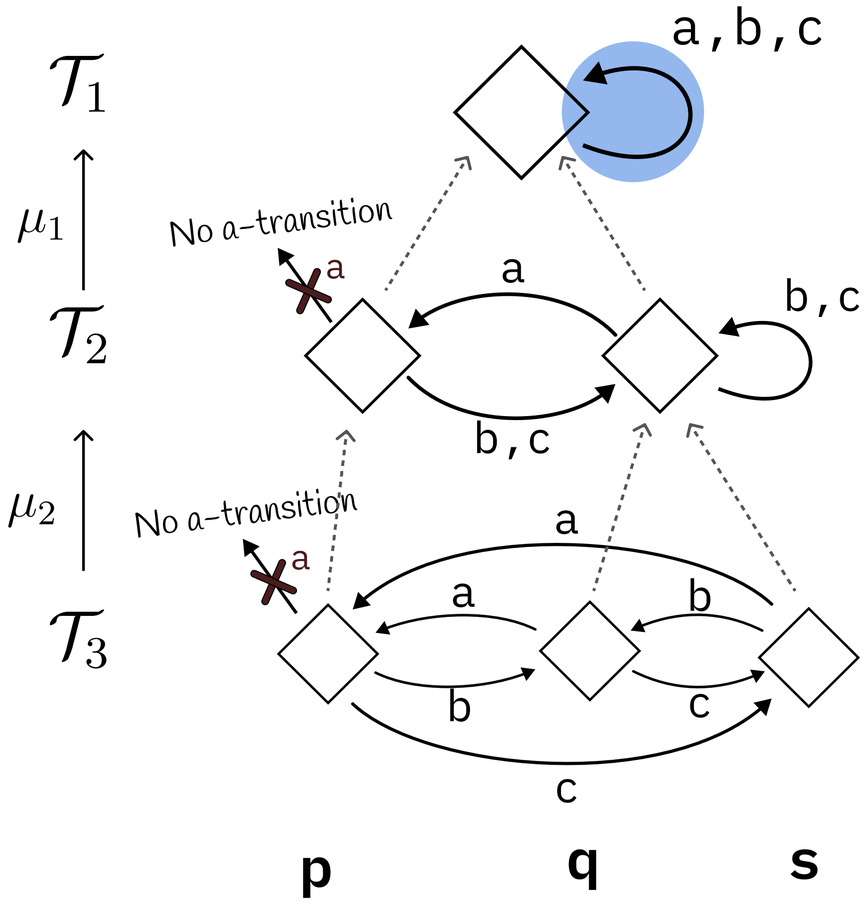


Alternating Parity Automaton

States: leaves

Transitions: given by the deepest $\mathcal{T}_i$ where the transition exists

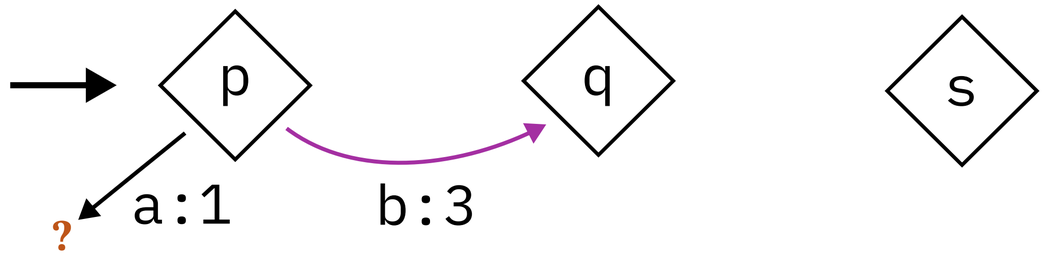


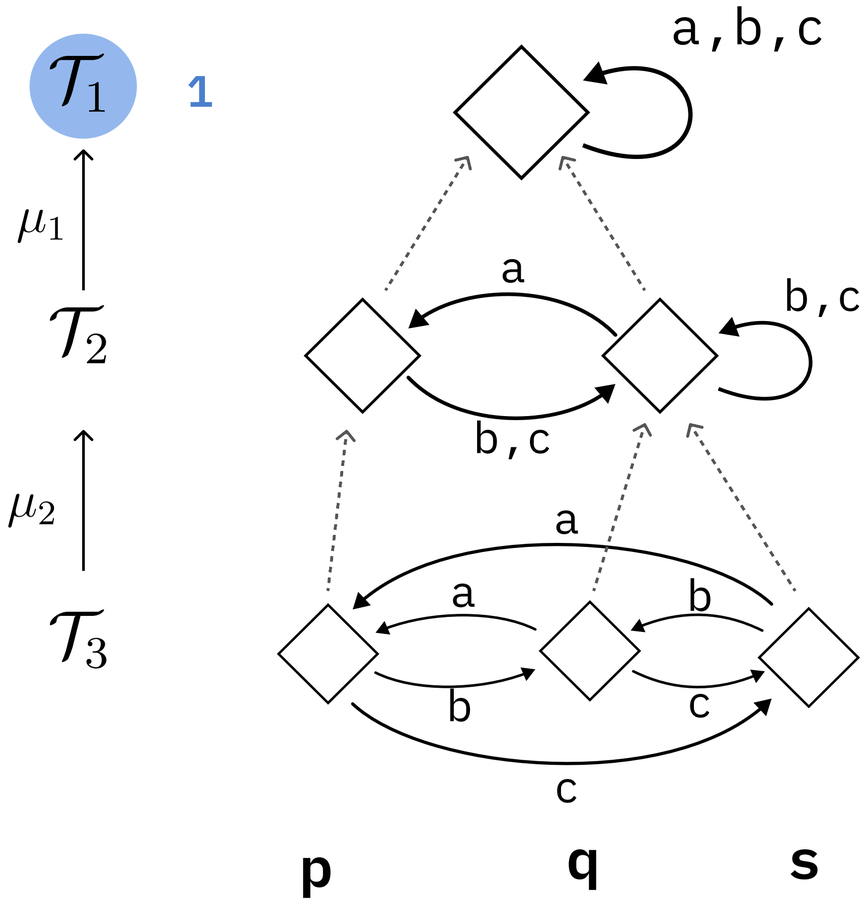


Alternating Parity Automaton

States: leaves

Transitions: given by the deepest $\mathcal{T}_i$ where the transition exists



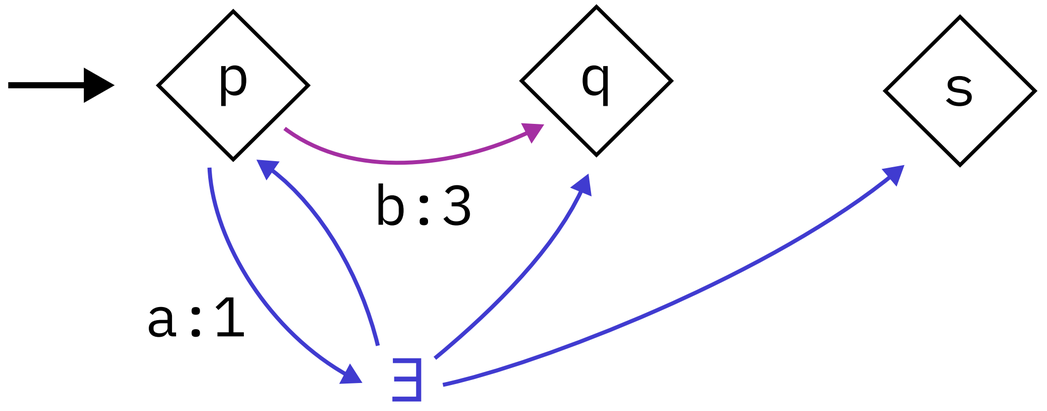


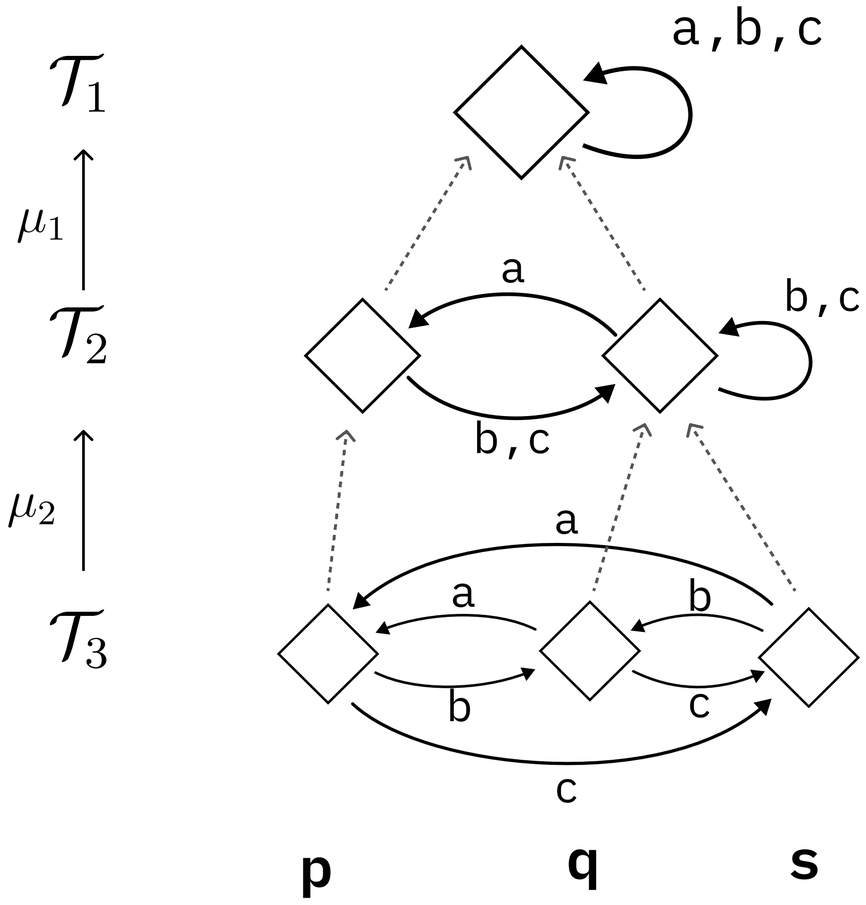
Odd → existential choice

Alternating Parity Automaton

States: leaves

Transitions: given by the deepest $\mathcal{T}_i$ where the transition exists



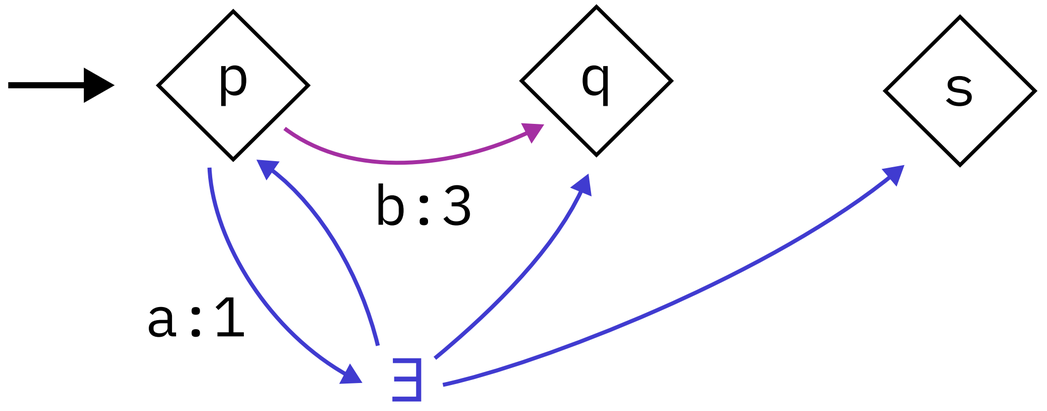


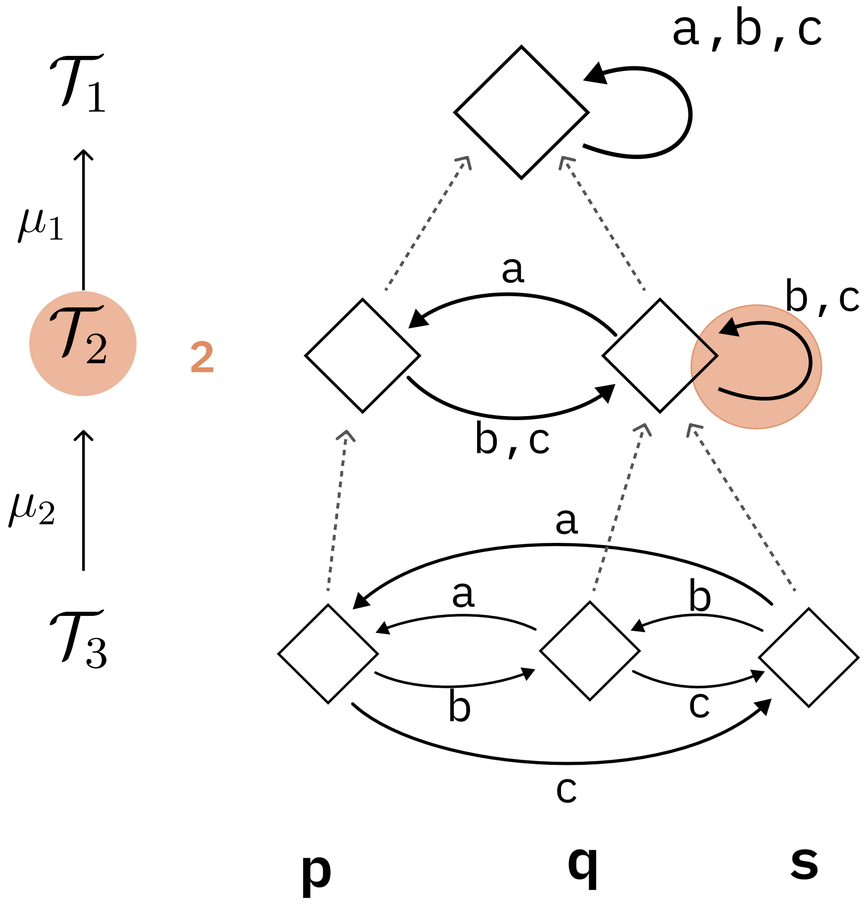
Odd → existential choice

Alternating Parity Automaton

States: leaves

Transitions: given by the deepest $\mathcal{T}_i$ where the transition exists

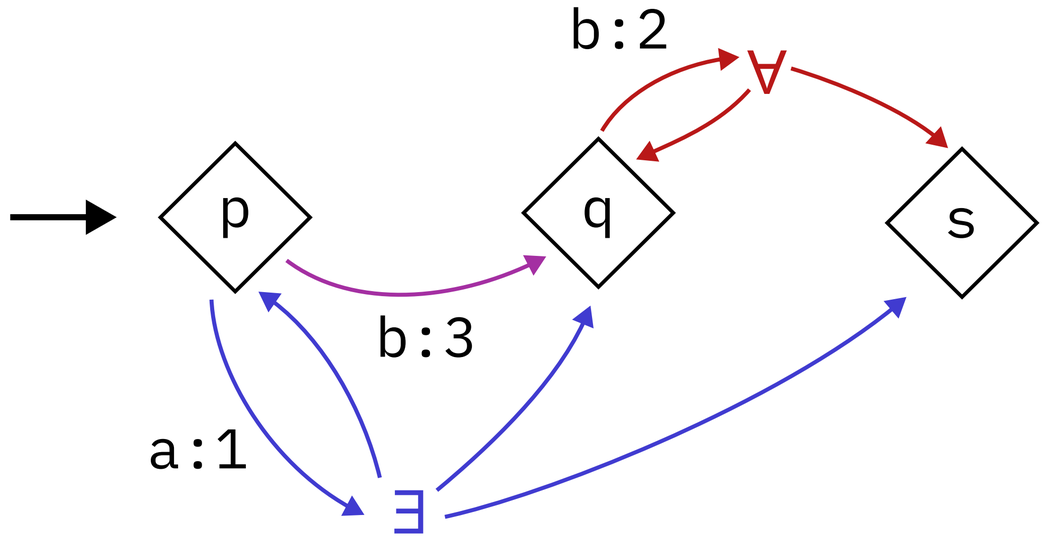




Alternating Parity Automaton

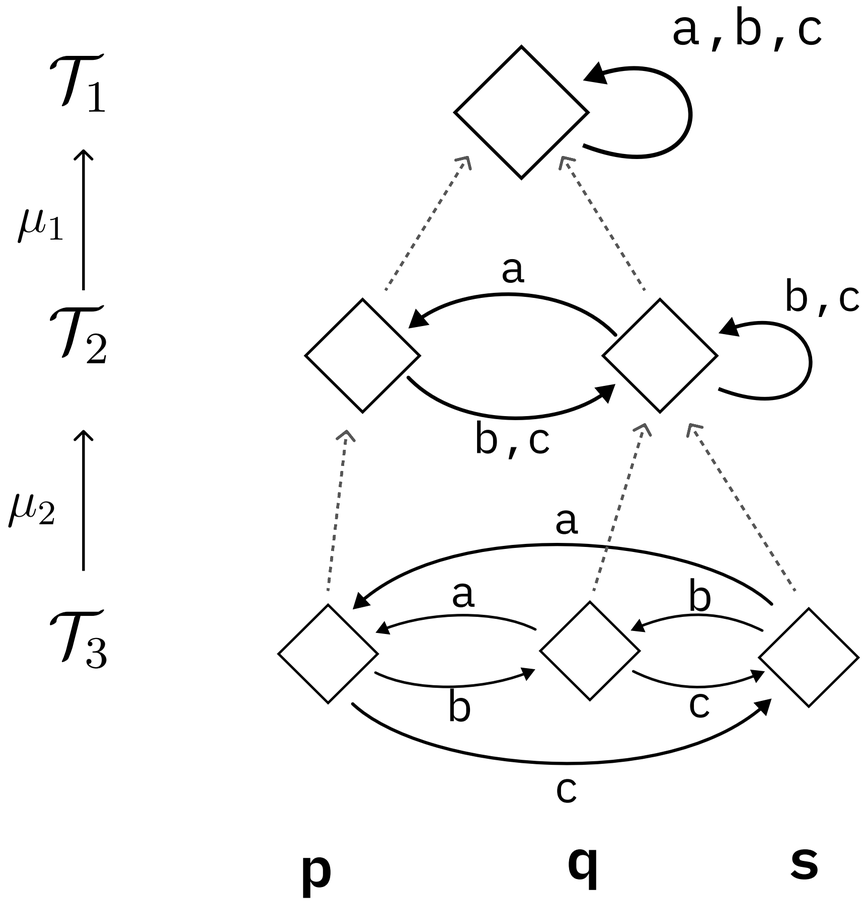
States: leaves

Transitions: given by the deepest $\mathcal{T}_i$ where the transition exists



Odd → existential choice

Even → universal choice



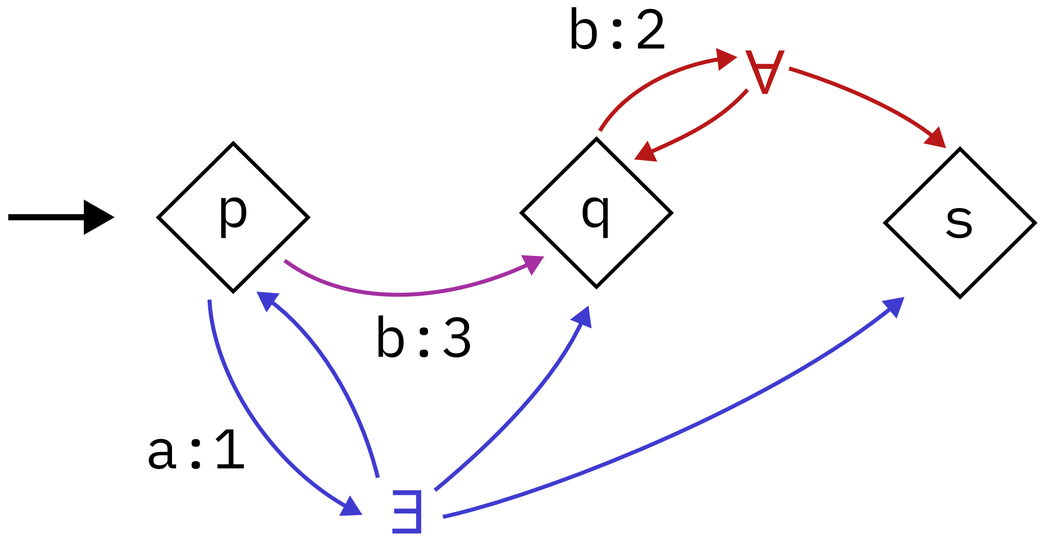
Odd → existential choice

Even → universal choice

Alternating Parity Automaton

States: leaves

Transitions: given by the deepest $\mathcal{T}_i$ where the transition exists



\+ $\mathcal{T}_1$ must correspond to residuals

Layered Automata

Formalism to represent a subclass of alternating parity automata

Definition

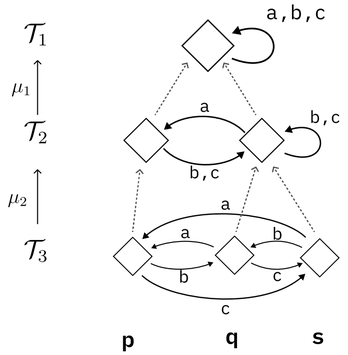
incomplete

Layered automaton: A sequence of deterministic transition systems with morphisms

$$\mathcal{T}_1 \xleftarrow{\mu_1} \mathcal{T}_2 \xleftarrow{\mu_2} \quad \dots \quad \mathcal{T}_{d-1} \xleftarrow{\mu_{d-1}} \mathcal{T}_d$$

$\mu : V \to V'$

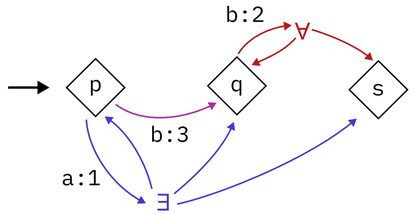
$p \xrightarrow{a}_{\text{in } \mathcal{T}} q \implies \mu(p) \xrightarrow{a}_{\text{in } \mathcal{T}'} \mu(q)$



Alternating Parity Automaton

States: leaves

Transitions: given by the deepest $\mathcal{T}_i$ where the transition exists



Odd → existential choice

Even → universal choice

\+ $\mathcal{T}_1$ must correspond to residuals

Layered Automata

Formalism to represent a subclass of alternating parity automata

Definition

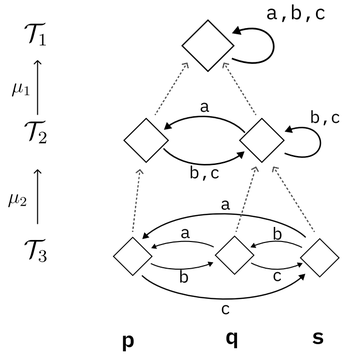
incomplete

Layered automaton: A sequence of deterministic transition systems with morphisms

$$\mathcal{T}_1 \xleftarrow{\mu_1} \mathcal{T}_2 \xleftarrow{\mu_2} \quad \dots \quad \mathcal{T}_{d-1} \xleftarrow{\mu_{d-1}} \mathcal{T}_d$$

$\mu : V \to V'$

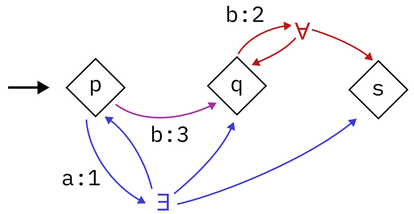
$p \xrightarrow{a}_{\text{in } \mathcal{T}} q \implies \mu(p) \xrightarrow{a}_{\text{in } \mathcal{T}'} \mu(q)$



Alternating Parity Automaton

States: leaves

Transitions: given by the deepest $\mathcal{T}_i$ where the transition exists



Odd → existential choice

Even → universal choice

\+ $\mathcal{T}_1$ must correspond to residuals

Properties

- Deterministic and minimal HD coBüchi automata are layered.

- Deterministic and minimal HD coBüchi automata are layered.

- THEOREM: All layered automata are history deterministic and 0-1 probabilistic.

- Deterministic and minimal HD coBüchi automata are layered.

- THEOREM: All layered automata are history deterministic and 0-1 probabilistic.

- THEOREM: Each ω-regular language admits a canonical minimal layered automaton.
  It can be computed in PTIME.

- Deterministic and minimal HD coBüchi automata are layered.

- THEOREM: All layered automata are history deterministic and 0-1 probabilistic.

- THEOREM: Each ω-regular language admits a canonical minimal layered automaton. (among layered automata)
  It can be computed in PTIME.

- Deterministic and minimal HD coBüchi automata are layered.

- THEOREM: All layered automata are history deterministic and 0-1 probabilistic.

- THEOREM: Each ω-regular language admits a canonical minimal layered automaton.
  It can be computed in PTIME.

among layered automata

from a given layered automaton

- Deterministic and minimal HD coBüchi automata are layered.

- THEOREM: All layered automata are history deterministic and 0-1 probabilistic.

- THEOREM: Each ω-regular language admits a canonical minimal layered automaton.
  It can be computed in PTIME.

among layered automata

from a given layered automaton

- Congruence-based characterisation of the minimal layered automaton.

Layered Automata

Formalism to represent a subclass of alternating parity automata

Definition

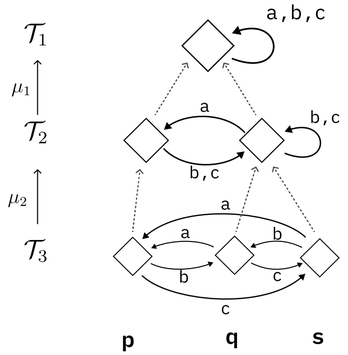
incomplete

Layered automaton: A sequence of deterministic transition systems with morphisms

$$\mathcal{T}_1 \xleftarrow{\mu_1} \mathcal{T}_2 \xleftarrow{\mu_2} \quad \dots \quad \mathcal{T}_{d-1} \xleftarrow{\mu_{d-1}} \mathcal{T}_d$$

$\mu : V \to V'$

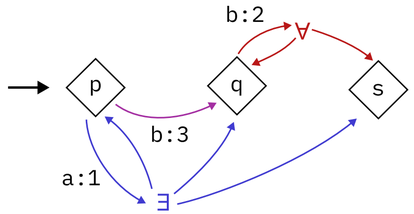
$p \xrightarrow{a}_{\text{in } \mathcal{T}} q \implies \mu(p) \xrightarrow{a}_{\text{in } \mathcal{T}'} \mu(q)$



Alternating Parity Automaton

States: leaves

Transitions: given by the deepest $\mathcal{T}_i$ where the transition exists



Odd → existential choice

Even → universal choice

\+ $\mathcal{T}_1$ must correspond to residuals

Properties

- Deterministic and minimal HD coBüchi automata are layered.
- THEOREM: All layered automata are history deterministic and 0-1 probabilistic.
- THEOREM: Each ω-regular language admits a canonical minimal layered automaton. (among layered automata)
  It can be computed in PTIME. (from a given layered automaton)
- Congruence-based characterisation of the minimal layered automaton.

Everybody loves automata

- Finite automata ⟷ MSO-logic *(Büchi, Elgot, Trakhtenbrot '60)*

Beautiful mathematical theory! — A minimal, canonical DFA for each regular language — **Efficiency in applications!**

- Minimisation in $O(n\log n)$ *(Hopcroft '71)*
- Automata learning in PTIME *(Gold '67, Angluin 87')*
- Simple characterization of FO-definability

⋮

Let's generalize to infinite words!

- ω-automata ⟷ MSO-logic *(Büchi '62)*
- Applications in verification and reactive synthesis

No unique minimal deterministic automata

Use of ω-automata very costly in applications!

- Minimisation is NP-complete *(Schewe '10, AbuRadi-Ehlers '25)*
- Costly determinization and boolean operations
- No efficient learning of ω-automata

Minimal HD coBüchi automata

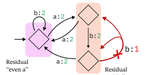
THEOREM *(Abu Radi-Kupferman '19)* — Using priorities {1,2}

History-deterministic (HD) coBüchi automata admit minimal, canonical automata, constructible in PTIME.

Properties of the minimal HD coBüchi automaton

A. If $p \xrightarrow{a:2} q$, then this is the only a-transition from p *(2-deterministic)*

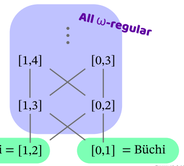
B. If $p \not\xrightarrow{a:2} q$, then also $p \xrightarrow{a:2} q_1, q_2, q_3, q_4$ to all states that are languages-equivalent to q *(1-saturated)*



Minimal HD coBüchi are great!

But they only recognize a small part of all ω-regular languages

Use alternating automata!



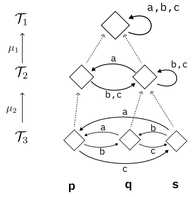
Instead, use Layered Automata

Formalism to represent a subclass of alternating parity automata

Definition

Layered automaton: A sequence of deterministic transition systems with morphisms (*incomplete*)

$$\mathcal{T}_1 \xleftarrow{\mu_1} \mathcal{T}_2 \xleftarrow{\mu_2} \dots \mathcal{T}_{d-1} \xleftarrow{\mu_{d-1}} \mathcal{T}_d$$



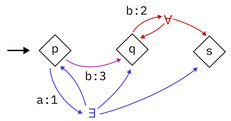
Odd → existential choice

Even → universal choice

Alternating Parity Automaton

States: leaves

Transitions: given by the deepest $\mathcal{T}_i$ where the transition exists



+ $\mathcal{T}_1$ must correspond to residuals

Properties

- Deterministic and minimal HD coBüchi automata are layered.
- THEOREM: All layered automata are history deterministic and 0-1 probabilistic.
- THEOREM: Each ω-regular language admits a canonical minimal layered automaton. (*among layered automata*)
  It can be computed in PTIME. (*from a given layered automaton*)
- Congruence-based characterisation of the minimal layered automaton.

Conclusions

Layered automata admit canonical, minimal automata, computable in polynomial time

Conclusions

Layered automata admit canonical, minimal automata, computable in polynomial time

Related work

Recently, other canonical representations of ω-regular languages have been introduced:

- Chains of coBüchi automata *(Schewe-Ehlers '22, Ehlers-Khalimov '24)*
- Rerailing automata *(Ehlers '25)*

Future work

Conjecture

The minimal layered automaton has a minimal number of states among all equivalent HD alternating parity automata.

Future work

Conjecture

The minimal layered automaton has a minimal number of states among all equivalent HD alternating parity automata.

Proved for Büchi and coBüchi!

Future work

Conjecture

Proved for Büchi and coBüchi!

The minimal layered automaton has a minimal number of states among all equivalent HD alternating parity automata.

- Passive and active learning of layered automata

Future work

Conjecture

Proved for Büchi and coBüchi!

The minimal layered automaton has a minimal number of states among all equivalent HD alternating parity automata.

- Passive and active learning of layered automata
- Implementation, boolean operations, translations from logic...

Future work

Conjecture

Proved for Büchi and coBüchi!

The minimal layered automaton has a minimal number of states among all equivalent HD alternating parity automata.

- Passive and active learning of layered automata
- Implementation, boolean operations, translations from logic...

Thanks for your attention!

Everybody loves automata

- Finite automata ⟷ MSO-logic *(Büchi, Elgot, Trakhtenbrot '60)*

Beautiful mathematical theory!

A minimal, canonical DFA for each regular language

Efficiency in applications!

- Minimisation in $O(n\log n)$ *(Hopcroft '71)*
- Automata learning in PTIME *(Gold '67, Angluin 87')*
- Simple characterization of FO-definability

⋮

Let's generalize to infinite words!

- ω-automata ⟷ MSO-logic *(Büchi '62)*
- Applications in verification and reactive synthesis

No unique minimal deterministic automata

Use of ω-automata very costly in applications!

- × Minimisation is NP-complete *(Schewe '10, AbuRadi-Ehlers '25)*
- × Costly determinization and boolean operations
- × No efficient learning of ω-automata

Minimal HD coBüchi automata

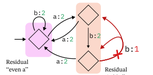
THEOREM *(Abu Radi-Kupferman '19)* Using priorities {1,2}

History-deterministic (HD) coBüchi automata admit minimal, canonical automata, constructible in PTIME.

Properties of the minimal HD coBüchi automaton

A. If $p \xrightarrow{a:2} q$, then this is the only a-transition from p *(2-deterministic)*

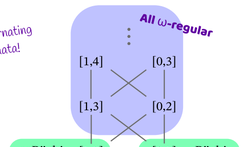
B. If $p \not\xrightarrow{a:2} q$, then also $p \xrightarrow{a:1} q_1, q_2, q_3, q_4$ to all states that are languages-equivalent to q *(1-saturated)*



Minimal HD coBüchi are great!

But they only recognize a small part of all ω-regular languages

Use alternating automata!



Conclusions

Layered automata admit canonical, minimal automata, computable in polynomial time

Related work

Recently, other canonical representations of ω-regular languages have been introduced:

- Chains of coBüchi automata *(Schewe-Ehlers '22, Ehlers-Khalimov '24)*
- Rerailing automata *(Ehlers '25)*

Future work

CONJECTURE Proved for Büchi and coBüchi!

The minimal layered automaton has a minimal number of states among all equivalent HD alternating parity automata.

- Passive and active learning of layered automata
- Implementation, boolean operations, translations from logic...

Thanks for your attention!

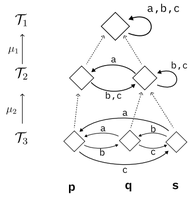
Instead, use ✨ *Layered Automata* ✨

Formalism to represent a subclass of alternating parity automata

Definition

Layered automaton: A sequence of deterministic transition systems with morphisms (incomplete)

$$\mathcal{T}_1 \xleftarrow{\mu_1} \mathcal{T}_2 \xleftarrow{\mu_2} \dots \quad \mathcal{T}_{d-1} \xleftarrow{\mu_{d-1}} \mathcal{T}_d$$



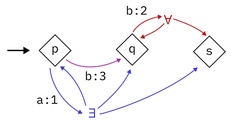
Odd → existential choice

Even → universal choice

Alternating Parity Automaton

States: leaves

Transitions: given by the deepest $\mathcal{T}_i$ where the transition exists



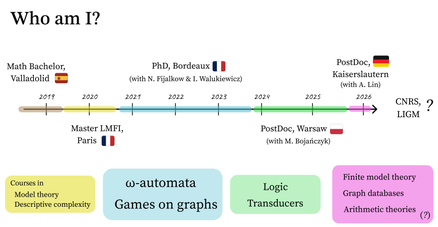
+ $\mathcal{T}_1$ must correspond to residuals

Properties

- Deterministic and minimal HD coBüchi automata are layered.
- THEOREM: All layered automata are history deterministic and 0-1 probabilistic.
- THEOREM: Each ω-regular language admits a canonical minimal layered automaton. (among layered automata)
  It can be computed in PTIME. (from a given layered automaton)
- Congruence-based characterisation of the minimal layered automaton.

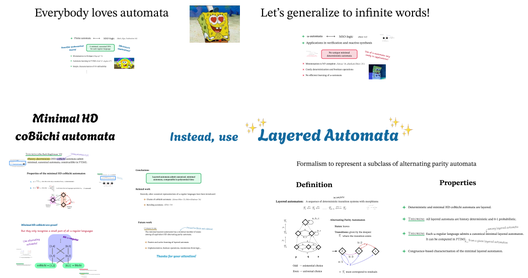
Layered Automata

A canonical model for ω-regular languages



Antonio Casares · RPTU Kaiserslautern

Joint work with: Christof Löding and Igor Walukiewicz



Séminaire BAAM, 4 November 2025