Some recent advances in register automata

Sławomir Lasota
University of Warsaw

CMCS 2022, 2022.04.02, Munich

I. Introduction to register automata

II. Some recent advances

Data languages = languages over infinite alphabet

Data languages = languages over infinite alphabet

- fix a countable infinite set $\mathbb{A} = \{\underline{1}, \underline{2}, \ldots\}$ of **data values (atoms)**

Data languages = languages over infinite alphabet

- fix a countable infinite set $\mathbb{A} = \{\underline{1}, \underline{2}, \ldots\}$ of **data values (atoms)**
- alphabet = $\mathbb{A}$ most often $\mathbb{A}$ × (finite set)

Data languages = languages over infinite alphabet

- fix a countable infinite set $\mathbb{A} = \{\underline{1}, \underline{2}, \ldots\}$ of **data values (atoms)**
- alphabet = $\mathbb{A}$ most often $\mathbb{A} \times$ (finite set)
- data words $w \in \mathbb{A}^*$

Data languages = languages over infinite alphabet

- fix a countable infinite set $\mathbb{A} = \{\underline{1}, \underline{2}, \ldots\}$ of **data values (atoms)**
- alphabet = $\mathbb{A}$ most often $\mathbb{A} \times$ (finite set)
- data words $w \in \mathbb{A}^*$
- data languages = subsets $L \subseteq \mathbb{A}^*$ **invariant** **(equivariant)** under data permutations:

  for every permutation $\pi : \mathbb{A} \to \mathbb{A}$,

  $a_1 \, a_2 \ldots a_n \in L$ implies $\pi(a_1) \, \pi(a_2) \ldots \pi(a_n) \in L$

Data languages = languages over infinite alphabet

- fix a countable infinite set $\mathbb{A} = \{\underline{1}, \underline{2}, \ldots\}$ of **data values (atoms)**
- alphabet = $\mathbb{A}$ most often $\mathbb{A} \times$ (finite set)
- data words $w \in \mathbb{A}^*$
- data languages = subsets $L \subseteq \mathbb{A}^*$ **(equivariant)** **invariant** under data permutations:

  for every permutation $\pi : \mathbb{A} \to \mathbb{A}$,

  $a_1 a_2 \ldots a_n \in L$ implies $\pi(a_1)\, \pi(a_2) \ldots \pi(a_n) \in L$

$$\pi(L) = L$$

Data languages = languages over infinite alphabet

- fix a countable infinite set $\mathbb{A} = \{\underline{1}, \underline{2}, \ldots\}$ of **data values (atoms)**
- alphabet = $\mathbb{A}$ most often $\mathbb{A} \times$ (finite set)
- data words $w \in \mathbb{A}^*$
- data languages = subsets $L \subseteq \mathbb{A}^*$ **invariant** **(equivariant)** under data permutations:

  for every permutation $\pi : \mathbb{A} \to \mathbb{A}$,

  $a_1\, a_2 \ldots a_n \in L$ implies $\pi(a_1)\, \pi(a_2) \ldots \pi(a_n) \in L$

$$\pi(L) = L$$

automorphisms of $(\mathbb{A}, =)$

action of the automorphism group

Data languages = languages over infinite alphabet

- fix a countable infinite set $\mathbb{A} = \{\underline{1}, \underline{2}, \ldots\}$ of **data values (atoms)**
- alphabet = $\mathbb{A}$ most often $\mathbb{A} \times$ (finite set)
- data words $w \in \mathbb{A}^*$

automorphisms of $(\mathbb{A}, =)$

(equivariant)

- data languages = subsets $L \subseteq \mathbb{A}^*$ **invariant** under data permutations:

for every permutation $\pi : \mathbb{A} \to \mathbb{A}$,

$a_1 a_2 \ldots a_n \in L$ implies $\pi(a_1)\, \pi(a_2) \ldots \pi(a_n) \in L$

$$\pi(L) = L$$

action of the automorphism group

Examples:

- first and last letter are equal

Data languages = languages over infinite alphabet

- fix a countable infinite set $\mathbb{A} = \{\underline{1}, \underline{2}, \ldots\}$ of **data values (atoms)**
- alphabet = $\mathbb{A}$ most often $\mathbb{A} \times$ (finite set)
- data words $w \in \mathbb{A}^*$

automorphisms of $(\mathbb{A}, =)$

(equivariant)

- data languages = subsets $L \subseteq \mathbb{A}^*$ **invariant** under data permutations:

for every permutation $\pi : \mathbb{A} \to \mathbb{A}$,

$a_1\, a_2 \ldots a_n \in L$ implies $\pi(a_1)\, \pi(a_2) \ldots \pi(a_n) \in L$

$$\pi(L) = L$$

action of the automorphism group

Examples:

- first and last letter are equal
- at most 3 different data values

Data languages = languages over infinite alphabet

- fix a countable infinite set $\mathbb{A} = \{\underline{1}, \underline{2}, \ldots\}$ of **data values (atoms)**
- alphabet = $\mathbb{A}$ most often $\mathbb{A} \times$ (finite set)
- data words $w \in \mathbb{A}^*$
- data languages = subsets $L \subseteq \mathbb{A}^*$ **invariant** **(equivariant)** under data permutations:

automorphisms of $(\mathbb{A}, =)$

for every permutation $\pi : \mathbb{A} \to \mathbb{A}$,

$a_1 \, a_2 \ldots a_n \in L$ implies $\pi(a_1) \, \pi(a_2) \ldots \pi(a_n) \in L$

$$\pi(L) = L$$

action of the automorphism group

Examples:

- first and last letter are equal
- at most 3 different data values
- all letters pairwise different

Data languages = languages over infinite alphabet

- fix a countable infinite set $\mathbb{A} = \{\underline{1}, \underline{2}, \ldots\}$ of **data values (atoms)**
- alphabet = $\mathbb{A}$ most often $\mathbb{A} \times$ (finite set)
- data words $w \in \mathbb{A}^*$

automorphisms of $(\mathbb{A}, =)$

(equivariant)

- data languages = subsets $L \subseteq \mathbb{A}^*$ **invariant** under data permutations:

  for every permutation $\pi : \mathbb{A} \to \mathbb{A}$,

  $a_1\, a_2 \ldots a_n \in L$ implies $\pi(a_1)\, \pi(a_2) \ldots \pi(a_n) \in L$

$$\pi(L) = L$$

action of the automorphism group

Examples:

- first and last letter are equal
- at most 3 different data values
- all letters pairwise different
- each two neighbouring letters are different

Register automata

- finite-memory automata Francez, Kaminski 1990
- history-dependent automata Pistore 1999
- **register automata** Neven, Schwentick, Vianu 2004
- nominal automata Bojańczyk, Klin, L. 2011
- automata with atoms Bojańczyk, Klin, L., Toruńczyk 2013
- orbit-finite automata
- FO-definable automata
- …

Register automata

- finite-memory automata — Francez, Kaminski 1990
- history-dependent automata — Pistore 1999
- **register automata** — Neven, Schwentick, Vianu 2004
- nominal automata — Bojańczyk, Klin, L. 2011
- automata with atoms — Bojańczyk, Klin, L., Toruńczyk 2013
- orbit-finite automata
- FO-definable automata
- …

- data automata
- symbolic automata
- pebble automata
- …

Register automata

- finite-memory automata — Francez, Kaminski 1990
- history-dependent automata — Pistore 1999
- **register automata** — Neven, Schwentick, Vianu 2004
- nominal automata — Bojańczyk, Klin, L. 2011
- automata with atoms — Bojańczyk, Klin, L., Toruńczyk 2013
- orbit-finite automata
- FO-definable automata
- …

- data automata
- symbolic automata
- pebble automata
- …

- FO over data words
- rigidly guarded MSO
- freeze LTL
- …

Register automata

- alphabet = $\mathbb{A}$
- states Q
- I, F $\subseteq$ Q subsets od initial and accepting states
- $\delta \subseteq Q \times \mathbb{A} \times Q$ transitions

Register automata

valuations of n registers storing data values

- alphabet = $\mathbb{A}$
- states Q **= (finite set of control locations C) × $(\mathbb{A} \cup \{\perp\})^n$**
- I, F $\subseteq$ ~~Q~~ C subsets od initial and accepting states
- $\delta \subseteq Q \times \mathbb{A} \times Q$ transitions

Register automata

- alphabet = $\mathbb{A}$
- states Q = **(finite set of control locations C)** $\times (\mathbb{A} \cup \{\perp\})^n$
- I, F $\subseteq$ ~~Q~~ C subsets od initial and accepting states
- $\delta \subseteq Q \times \mathbb{A} \times Q$ transitions

valuations of n registers storing data values

initial register valuation $(\perp, \perp, \ldots, \perp)$

Register automata

- alphabet = $\mathbb{A}$
- states Q **= (finite set of control locations C) × $(\mathbb{A} \cup \{\bot\})^n$**
- I, F ⊆ ~~Q~~ C subsets od initial and accepting states
- $\delta \subseteq Q \times \mathbb{A} \times Q$ transitions

valuations of n registers storing data values

initial register valuation $(\bot, \bot, \ldots, \bot)$

transitions compare input to registers and update registers

Register automata - transitions

- **syntactic** presentation:

- **semantic** presentation:

Register automata - transitions

- **syntactic** presentation:

finite set of rules of the form (c, φ, c'),

$\varphi(x_1 \ldots x_n\ y\ x'_1 \ldots x'_n)$ is a Boolean combination of equalities - **constraint**

- **semantic** presentation:

Register automata - transitions

- **syntactic** presentation:

finite set of rules of the form (c, φ, c'),

$\varphi(x_1 \ldots x_n\ y\ x'_1 \ldots x'_n)$ is a Boolean combination of equalities - **constraint**

a rule induces a transition

$(c, a_1 \ldots a_n) \xrightarrow{b} (c', a'_1 \ldots a'_n)$

whenever $\varphi(a_1 \ldots a_n\ b\ a'_1 \ldots a'_n)$

- **semantic** presentation:

Register automata - transitions

- **syntactic** presentation:

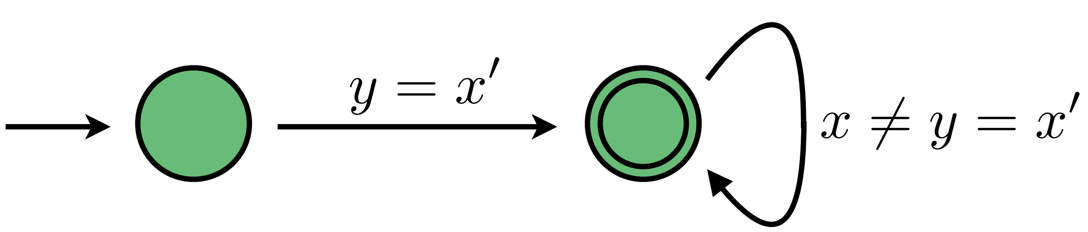
finite set of rules of the form (c, φ, c'),

$\varphi(x_1 \ldots x_n\ y\ x'_1 \ldots x'_n)$ is a Boolean combination of equalities - **constraint**

a rule induces a transition

$(c, a_1 \ldots a_n) \xrightarrow{b} (c', a'_1 \ldots a'_n)$

whenever $\varphi(a_1 \ldots a_n\ b\ a'_1 \ldots a'_n)$



- **semantic** presentation:

Register automata - transitions

- **syntactic** presentation:

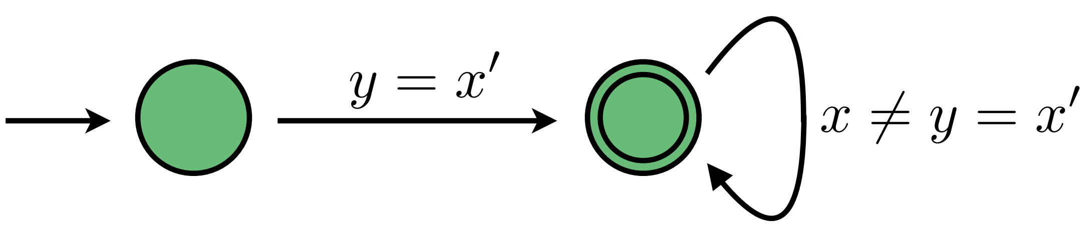
finite set of rules of the form (c, φ, c'),

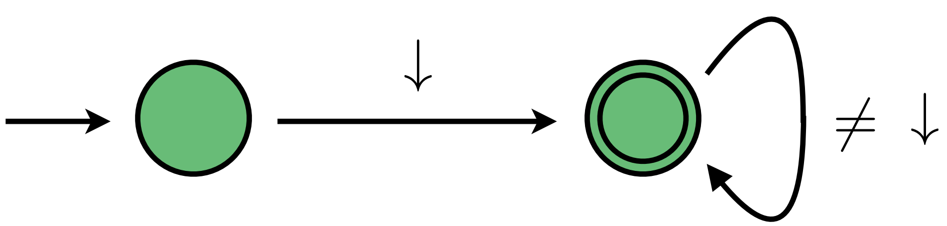
$\varphi(x_1 \ldots x_n \; y \; x'_1 \; \ldots \; x'_n)$ is a Boolean combination of equalities - **constraint**

a rule induces a transition

$(c, a_1 \ldots a_n) \xrightarrow{b} (c', a'_1 \ldots a'_n)$

whenever $\varphi(a_1 \ldots a_n \; b \; a'_1 \; \ldots a'_n)$





- **semantic** presentation:

Register automata - transitions

- **syntactic** presentation:

finite set of rules of the form (c, φ, c'),

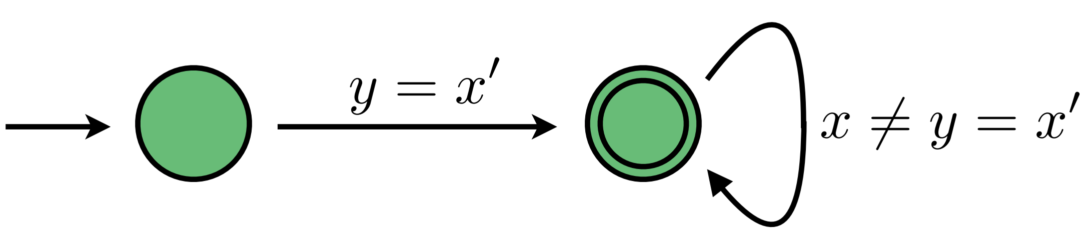
$\varphi(x_1 \ \dots \ x_n \ y \ x'_1 \ \dots \ x'_n)$ is a Boolean combination of equalities - **constraint**

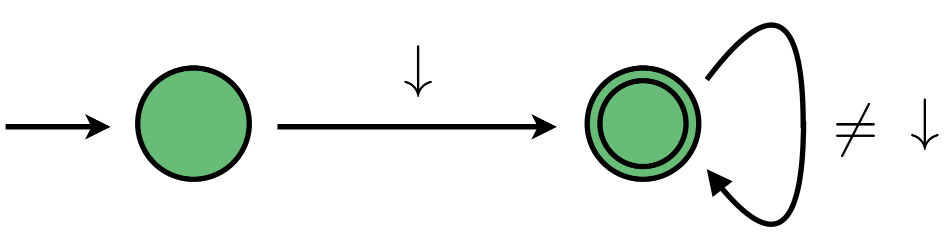
a rule induces a transition

$(c, a_1 \dots a_n) \xrightarrow{b} (c', a'_1 \dots a'_n)$

whenever $\varphi(a_1 \dots a_n \ b \ a'_1 \ \dots a'_n)$

- **semantic** presentation:





"each two neighbouring letters are different and the word is nonempty"

Register automata - transitions

- **syntactic** presentation:

  finite set of rules of the form (c, φ, c'),

  $\varphi(x_1 \ldots x_n\ y\ x'_1\ \ldots\ x'_n)$ is a Boolean combination of equalities - **constraint**

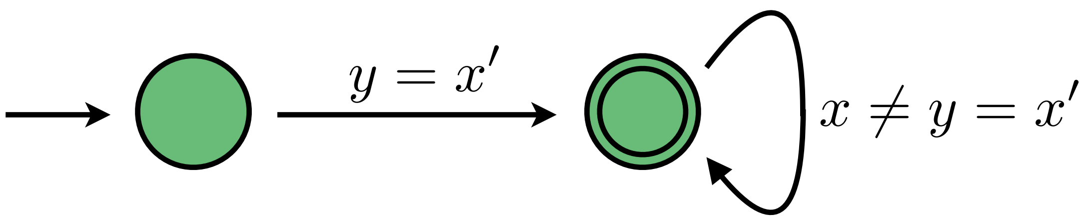
  a rule induces a transition

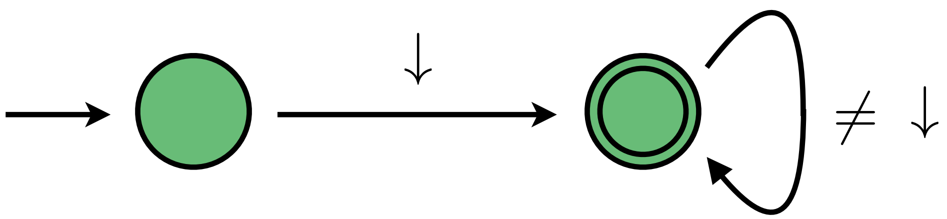
  $(c, a_1 \ldots a_n) \xrightarrow{b} (c', a'_1 \ldots a'_n)$

  whenever $\varphi(a_1 \ldots a_n\ b\ a'_1\ \ldots a'_n)$

- **semantic** presentation:

  $\delta \subseteq Q \times \mathbb{A} \times Q$ **invariant** under data permutations





"each two neighbouring letters are different and the word is nonempty"

Register automata - transitions

- **syntactic** presentation:

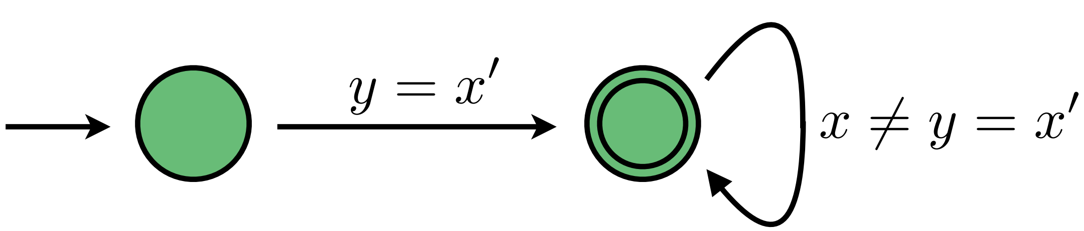
finite set of rules of the form (c, φ, c'),

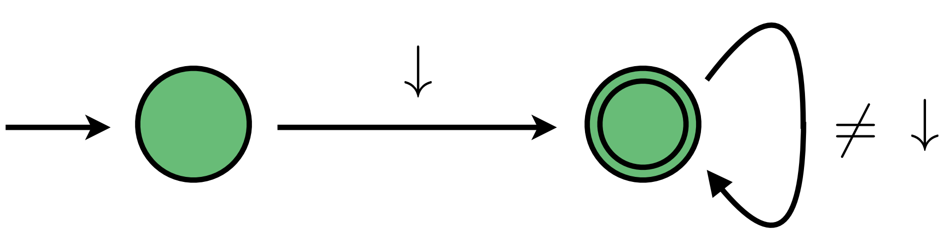
$\varphi(x_1 \dots x_n\ y\ x'_1\ \dots\ x'_n)$ is a Boolean combination of equalities - **constraint**

a rule induces a transition

$(c, a_1 \dots a_n) \xrightarrow{b} (c', a'_1 \dots a'_n)$

whenever $\varphi(a_1 \dots a_n\ b\ a'_1\ \dots a'_n)$





- **semantic** presentation:

$\delta \subseteq Q \times \mathbb{A} \times Q$ **invariant** under data permutations

for every permutation $\pi : \mathbb{A} \to \mathbb{A}$,

$(c, a_1 \dots a_n) \xrightarrow{b} (c', a'_1 \dots a'_n)$

$\Downarrow$

$(c, \pi(a_1) \dots \pi(a_n)) \xrightarrow{\pi(b)} (c', \pi(a'_1) \dots \pi(a'_n))$

$\pi(\perp) = \perp$

"each two neighbouring letters are different and the word is nonempty"

Register automata - transitions

- **syntactic** presentation:

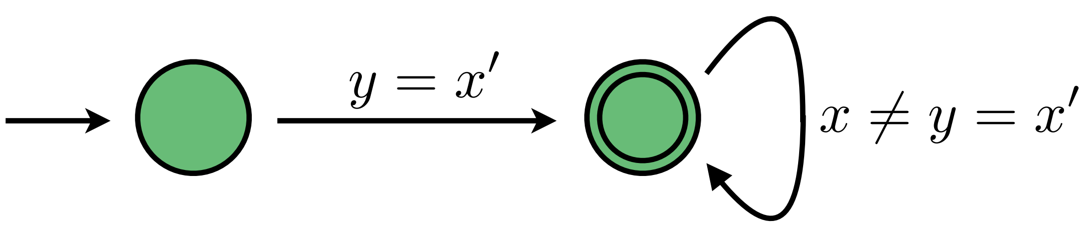
finite set of rules of the form (c, φ, c'),

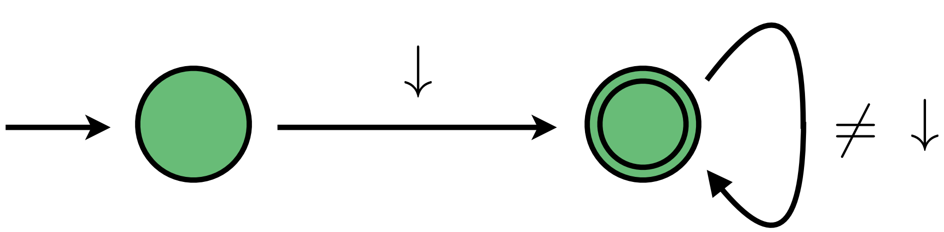
$\varphi(x_1 \dots x_n\ y\ x'_1 \dots x'_n)$ is a Boolean combination of equalities - **constraint**

a rule induces a transition

$(c, a_1 \dots a_n) \xrightarrow{b} (c', a'_1 \dots a'_n)$

whenever $\varphi(a_1 \dots a_n\ b\ a'_1 \dots a$





- **semantic** presentation:

$\delta \subseteq Q \times \mathbb{A} \times Q$ **invariant** under data permutations

for every permutation $\pi : \mathbb{A} \to \mathbb{A}$,

$(c, a_1$ $) \xrightarrow{b} (c', a'_1 \dots a'_n)$

$\Downarrow$

$\pi(a_1) \dots \pi(a_n)) \xrightarrow{\pi(b)} (c', \pi(a'_1) \dots \pi(a'_n))$

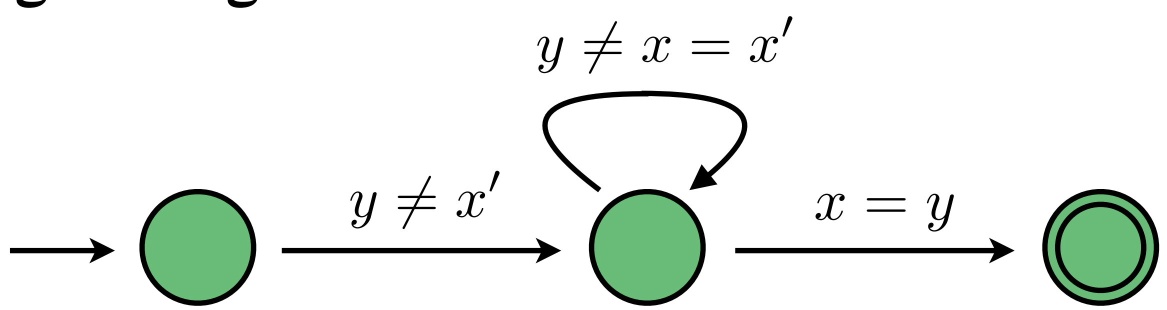
$\pi(\perp) = \perp$

"each two neighbouring letters are different and the word is nonempty"

the presentations are equivalent

Nondeterminism in register automata

- **guessing:**

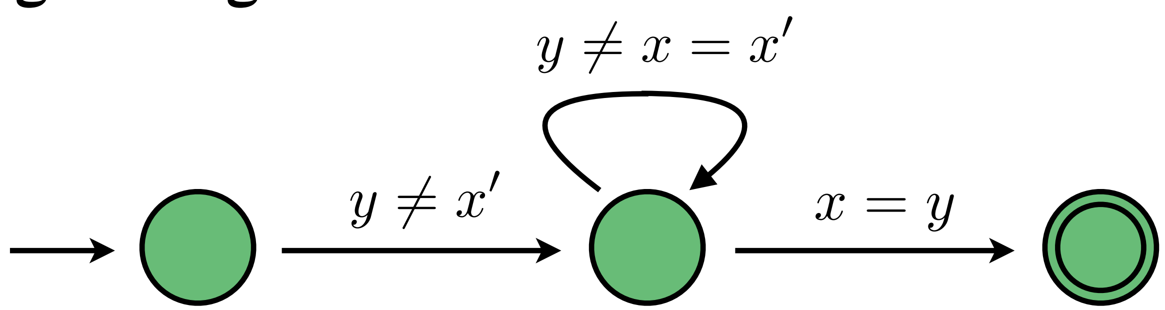


"last data does not appear earlier"

(unambiguous - unique accepting runs)

Nondeterminism in register automata

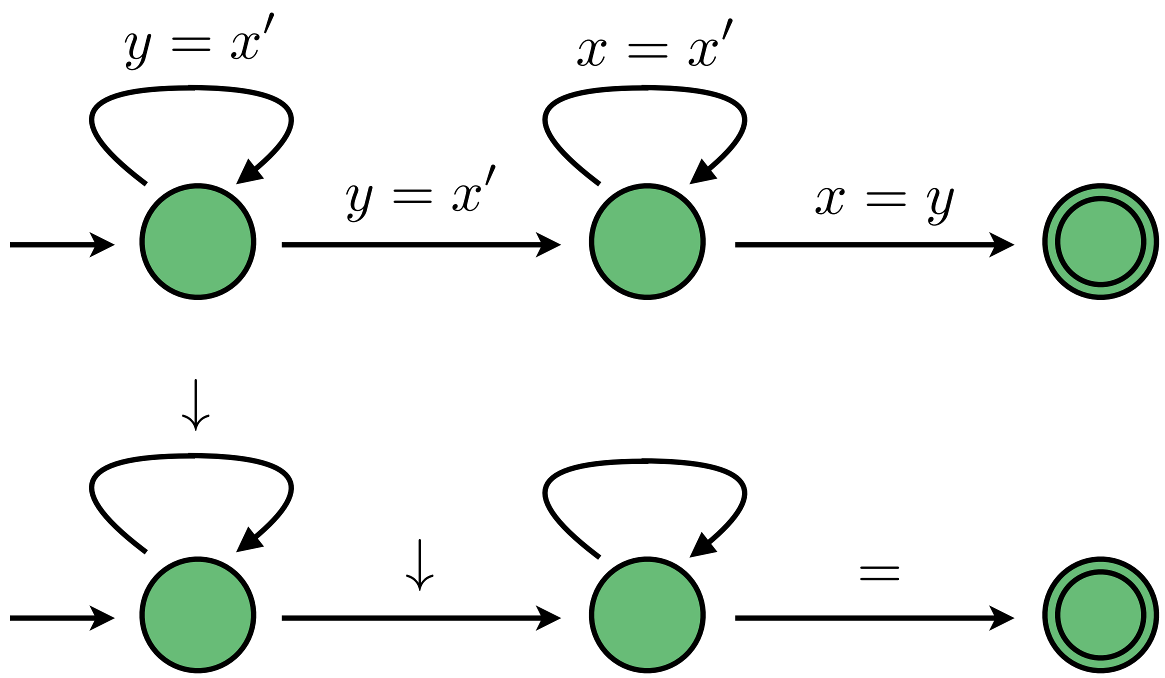
- **guessing:**



"last data does not appear earlier"

(unambiguous - unique accepting runs)

- **no guessing:**

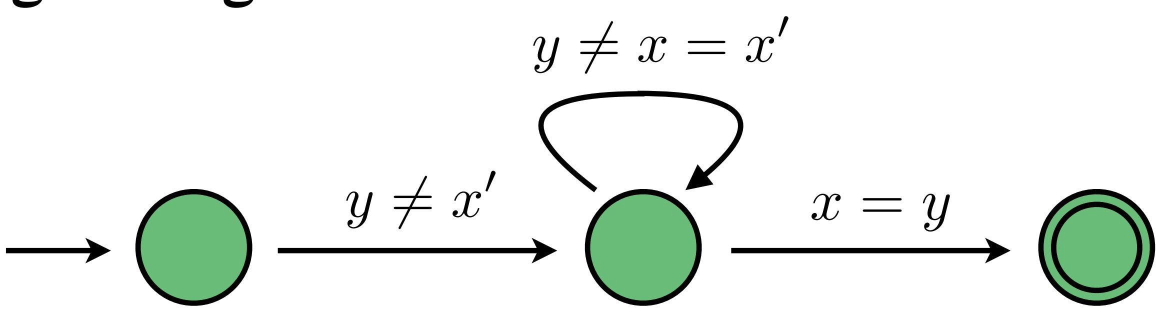


"last data appears earlier"

(can be made unambiguous)

Nondeterminism in register automata

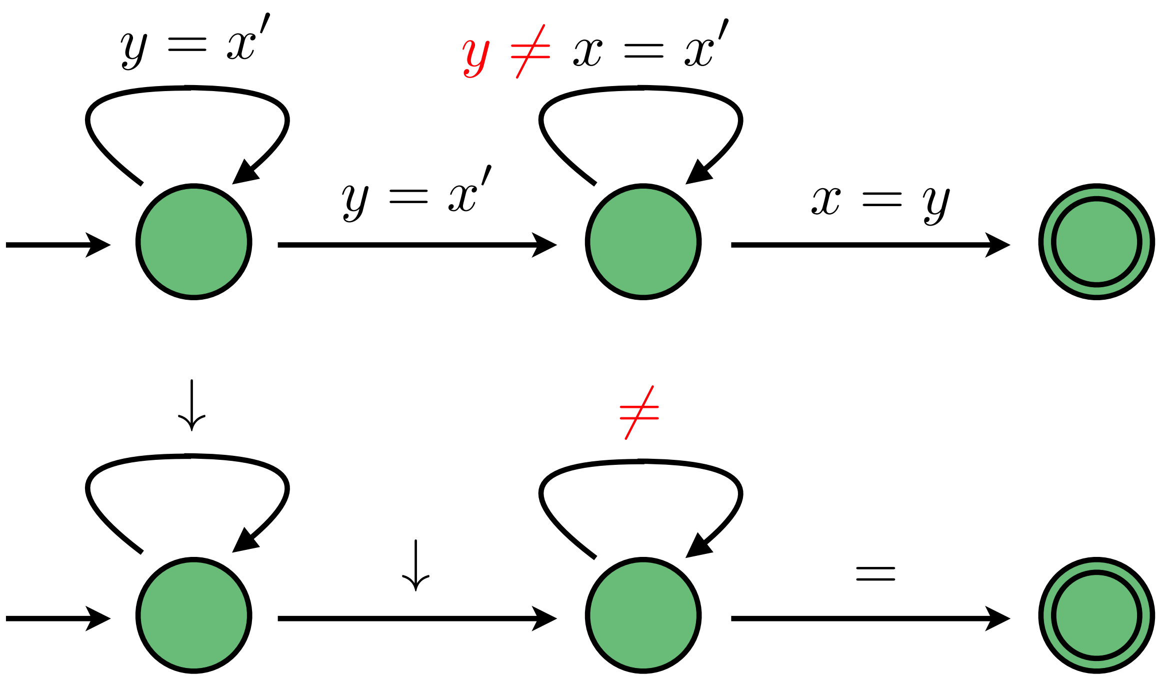
- **guessing:**



"last data does not appear earlier"

(unambiguous - unique accepting runs)

- **no guessing:**

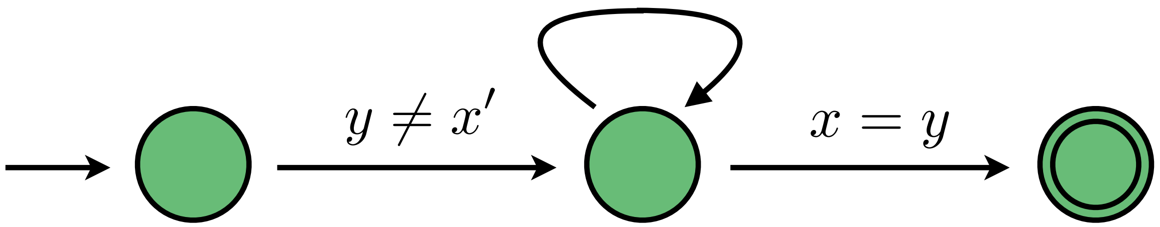


"last data appears earlier"

(can be made unambiguous)

Nondeterminism in register automata

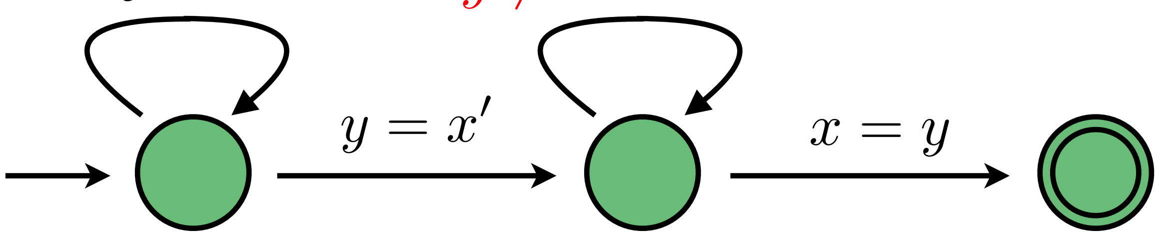
- **guessing:**

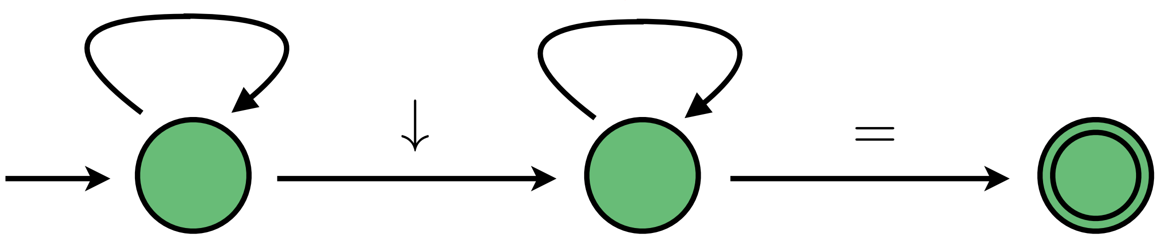


"last data does not appear earlier"

(unambiguous - unique accepting runs)

- **no guessing:**





"last data appears earlier"

(can be made unambiguous)

DRA $\subseteq$ URA $\subseteq$ NRA $\subseteq$ ARA

NRA $\supseteq$ NRA without guessing

ARA $\supseteq$ ARA without guessing

Orbit-finite sets

X and Y are **in the same orbit** if $X = \pi(Y)$ for some data permutation π

Orbit-finite sets

X and Y are **in the same orbit** if $X = \pi(Y)$
for some data permutation π

Examples:

- n-tuples of data values $\mathbb{A}^n$ Bell number B_n

Orbit-finite sets

X and Y are **in the same orbit** if $X = \pi(Y)$ for some data permutation π

Examples:

- n-tuples of data values $\mathbb{A}^n$ — Bell number B_n
- register valuations $(\mathbb{A} \cup \{\bot\})^n$

Orbit-finite sets

X and Y are **in the same orbit** if $X = \pi(Y)$
for some data permutation π

Examples:

- n-tuples of data values $\mathbb{A}^n$ Bell number B_n
- register valuations $(\mathbb{A} \cup \{\perp\})^n$
- states $C \times (\mathbb{A} \cup \{\perp\})^n$

Orbit-finite sets

X and Y are **in the same orbit** if $X = \pi(Y)$
for some data permutation π

Examples:

- n-tuples of data values $\mathbb{A}^n$ — Bell number B_n
- register valuations $(\mathbb{A} \cup \{\perp\})^n$
- states $C \times (\mathbb{A} \cup \{\perp\})^n$
- non-repeating n-tuples $\mathbb{A}^{(n)}$ — *1*

Orbit-finite sets

X and Y are **in the same orbit** if $X = \pi(Y)$ for some data permutation π

Examples:

- n-tuples of data values $\mathbb{A}^n$ — Bell number B_n
- register valuations $(\mathbb{A} \cup \{\perp\})^n$
- states $C \times (\mathbb{A} \cup \{\perp\})^n$
- non-repeating n-tuples $\mathbb{A}^{(n)}$ — *1*
- n-sets of data values $P_n(\mathbb{A})$ — *1*

Orbit-finite sets

> X and Y are **in the same orbit** if $X = \pi(Y)$ for some data permutation π

Examples:

- n-tuples of data values $\mathbb{A}^n$ — Bell number B_n
- register valuations $(\mathbb{A} \cup \{\bot\})^n$
- states $C \times (\mathbb{A} \cup \{\bot\})^n$
- non-repeating n-tuples $\mathbb{A}^{(n)}$ — *1*
- n-sets of data values $P_n(\mathbb{A})$ — *1*
- finite words $\mathbb{A}^*$ — ∞

Orbit-finite automata

X and Y are **in the same orbit** if $X = \pi(Y)$ for some data permutation π

- Nondeterministic **orbit-finite automata NOFA**
  - alphabet = $\mathbb{A}$
  - Q - an **orbit-finite** invariant set of states
  - I, F $\subseteq$ Q - invariant subsets od initial and accepting states
  - $\delta \subseteq Q \times \mathbb{A} \times Q$ invariant set of transitions

Orbit-finite automata

X and Y are **in the same orbit** if $X = \pi(Y)$ for some data permutation π

- Nondeterministic **orbit-finite automata NOFA**
  - alphabet = $\mathbb{A}$
  - Q - an **orbit-finite** invariant set of states
  - I, F $\subseteq$ Q - invariant subsets od initial and accepting states
  - $\delta \subseteq Q \times \mathbb{A} \times Q$ invariant set of transitions

Theorem (Bojańczyk, Klin, L. 2014):
NOFA = NRA
DOFA = DRA

Orbit-finite automata

> X and Y are **in the same orbit** if $X = \pi(Y)$ for some data permutation π

- Nondeterministic **orbit-finite automata NOFA**
  - alphabet = $\mathbb{A}$
  - Q - an **orbit-finite** invariant set of states
  - I, F $\subseteq$ Q - invariant subsets od initial and accepting states
  - $\delta \subseteq Q \times \mathbb{A} \times Q$ invariant set of transitions

Theorem (Bojańczyk, Klin, L. 2014):
NOFA = NRA
DOFA = DRA

- Arbitrary **orbit-finite alphabets**

Orbit-finite automata

X and Y are **in the same orbit** if $X = \pi(Y)$ for some data permutation π

- Nondeterministic **orbit-finite automata NOFA**
  - alphabet = $\mathbb{A}$
  - Q - an **orbit-finite** invariant set of states
  - I, F $\subseteq$ Q - invariant subsets od initial and accepting states
  - $\delta \subseteq Q \times \mathbb{A} \times Q$ invariant set of transitions

Theorem (Bojańczyk, Klin, L. 2014):
NOFA = NRA
DOFA = DRA

- Arbitrary **orbit-finite alphabets**
- Richer structure of data values than ($\mathbb{A}$, =)

e.g. ($\mathbb{Q}$, <)
any homogeneous/oligomorphic one

Non-robustness of register automata

- ARA
- NRA
- URA
- DRA

Non-robustness of register automata

- ARA
- NRA
- URA
- DRA

} with guessing or not

Non-robustness of register automata

1-way or 2-way

- ARA
- NRA
- URA
- DRA

with guessing or not

Non-robustness of register automata

1-way or 2-way {

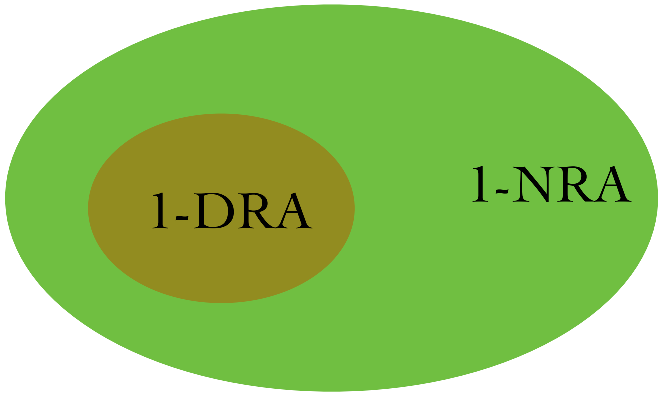
- ARA
- NRA
- URA
- DRA

} with guessing or not

Non-robustness of register automata

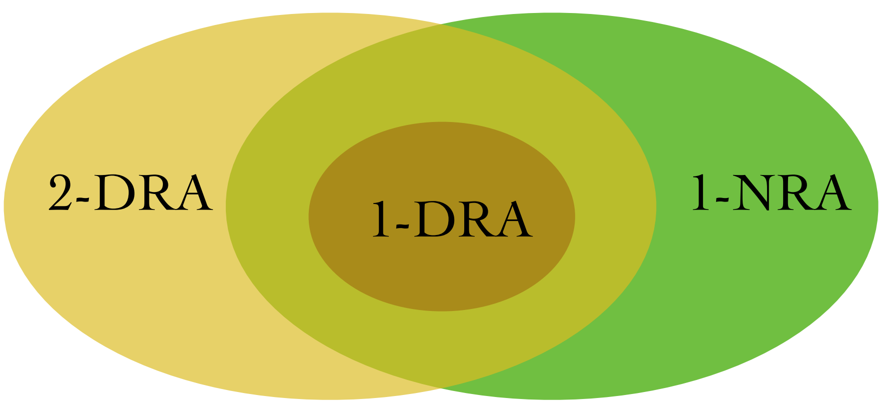
1-way or 2-way {
- ARA
- NRA
- URA
- DRA

} with guessing or not



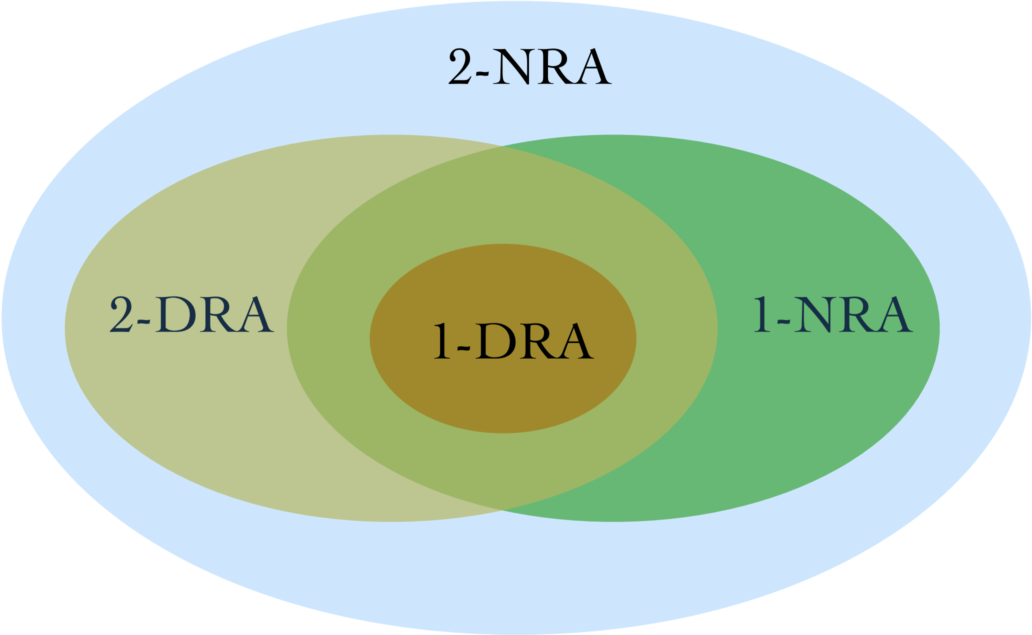
Non-robustness of register automata

1-way or 2-way { • ARA • NRA • URA • DRA } with guessing or not



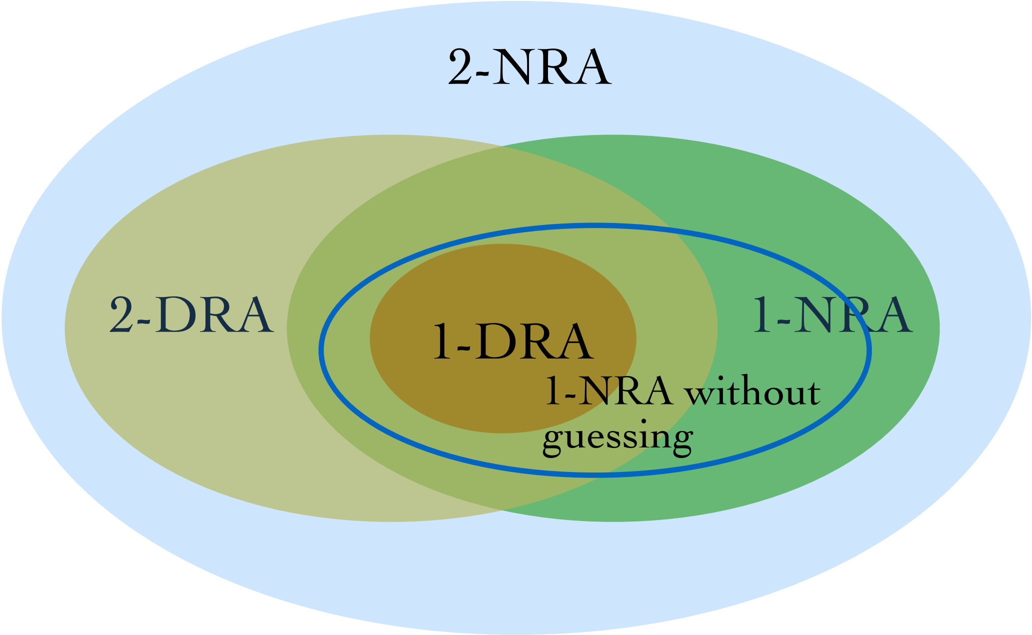
Non-robustness of register automata

1-way or 2-way { • ARA • NRA • URA • DRA } with guessing or not



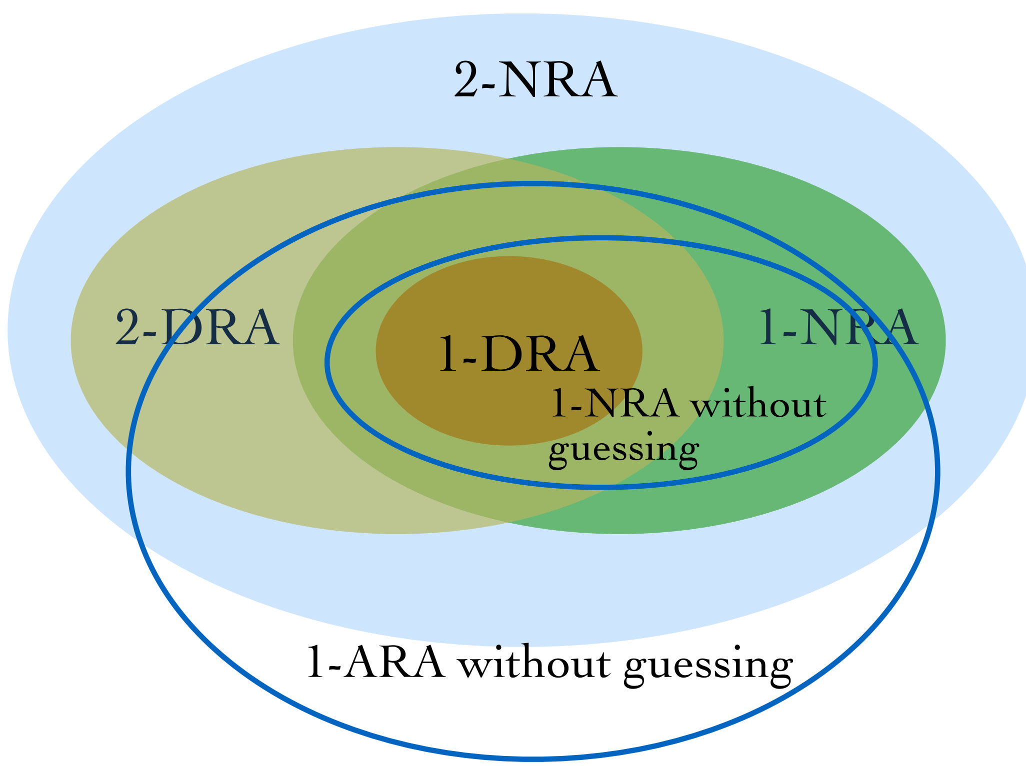
Non-robustness of register automata

1-way or 2-way { • ARA • NRA • URA • DRA } with guessing or not



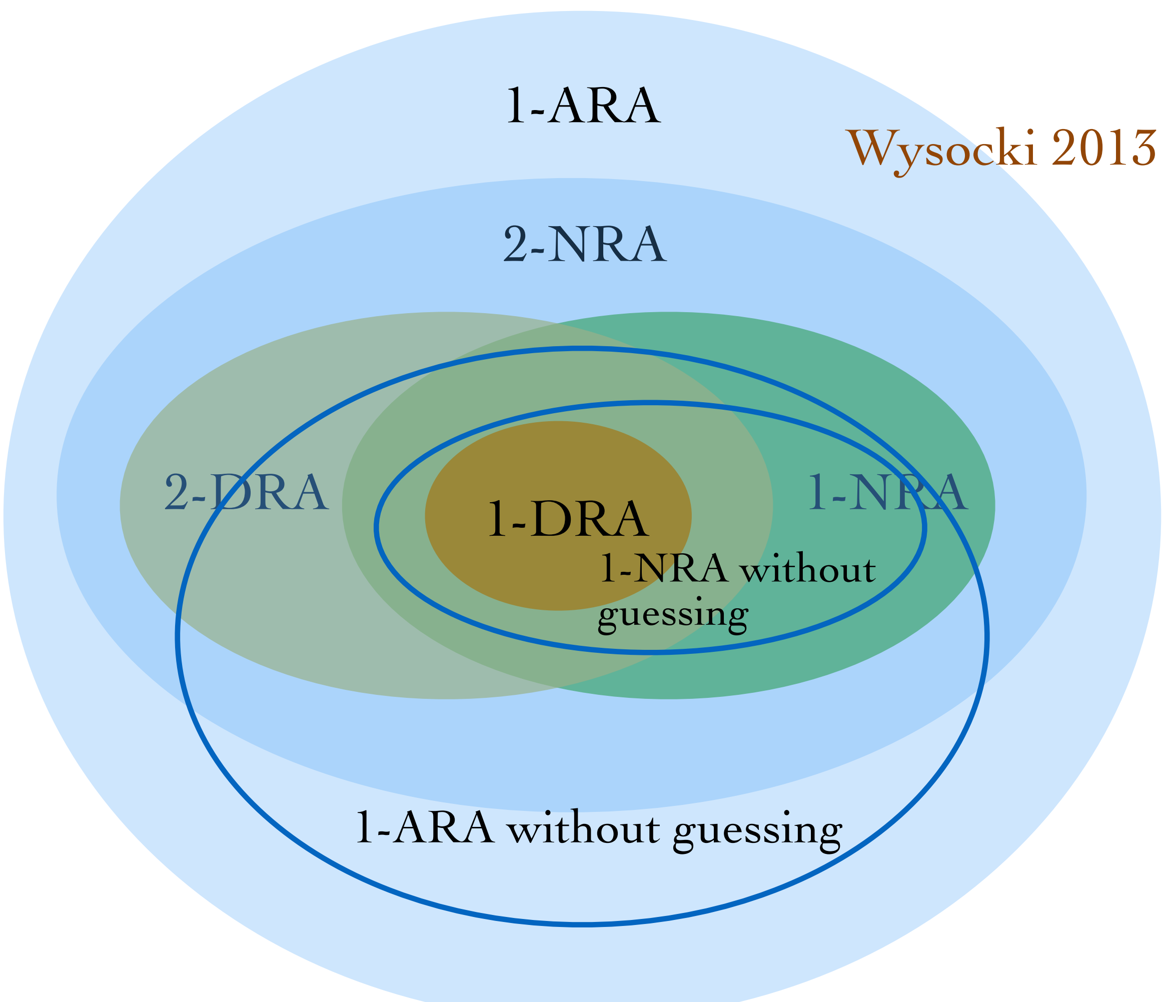
Non-robustness of register automata

1-way or 2-way { • ARA • NRA • URA • DRA } with guessing or not



Non-robustness of register automata

1-way or 2-way { • ARA • NRA • URA • DRA } with guessing or not

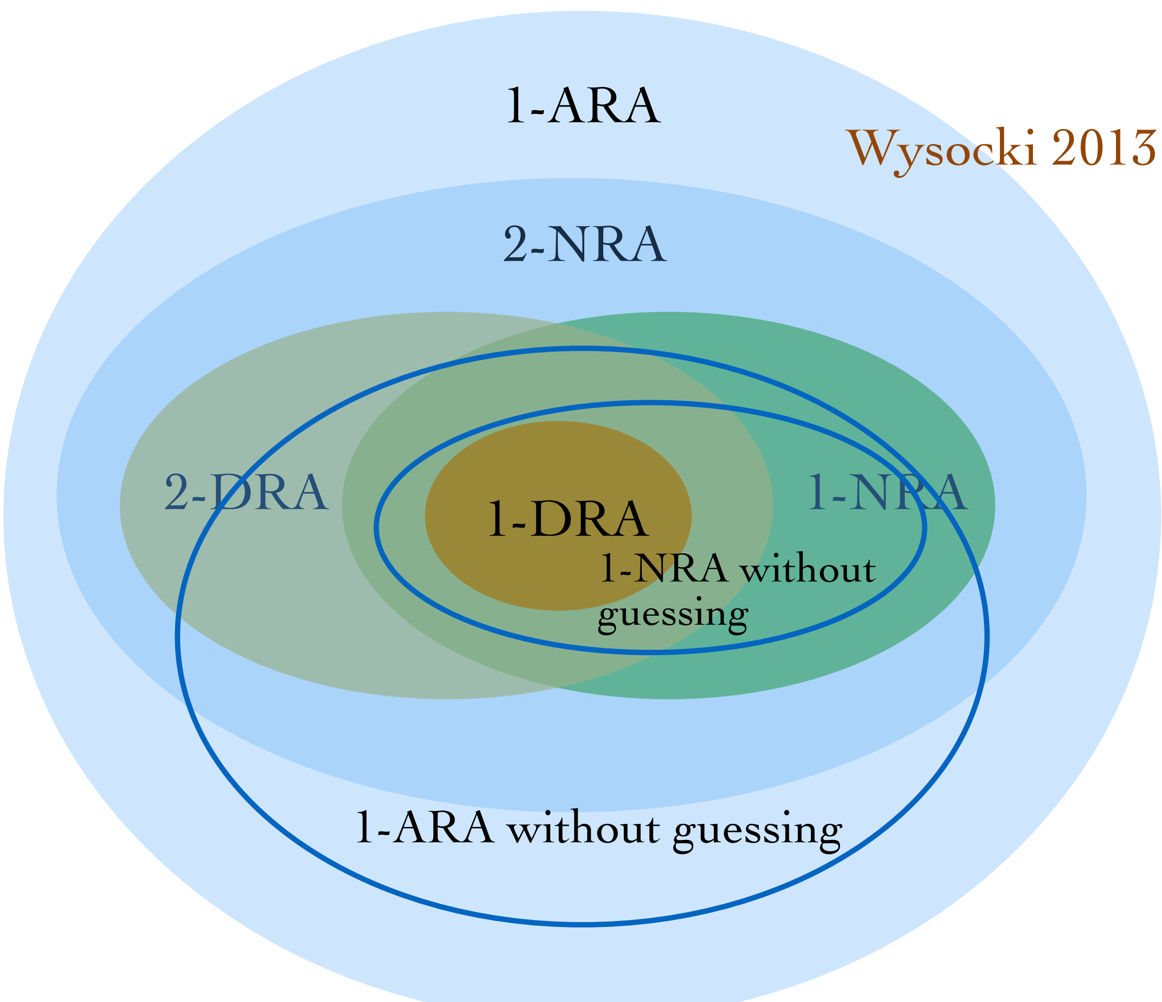


Non-robustness of register automata

1-way or 2-way {
- ARA
- NRA
- URA
- DRA
} with guessing or not

DRA = left congruence of orbit-finite index

Bojańczyk, Klin, L. 2014

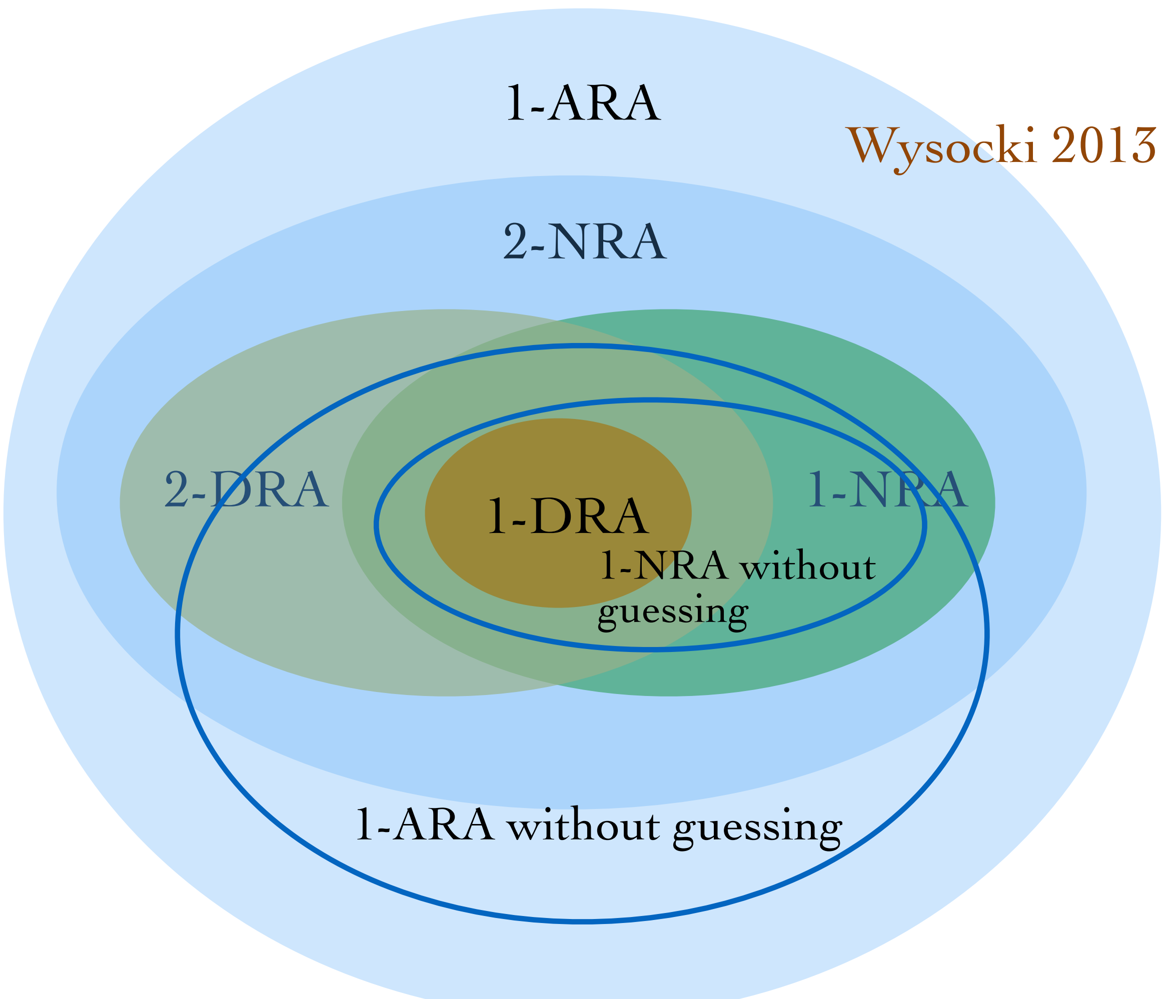


Non-robustness of register automata

1-way or 2-way { • ARA • NRA • URA • DRA } with guessing or not

DRA = left congruence of orbit-finite index ⊃ orbit-finite monoids

Bojańczyk, Klin, L. 2014



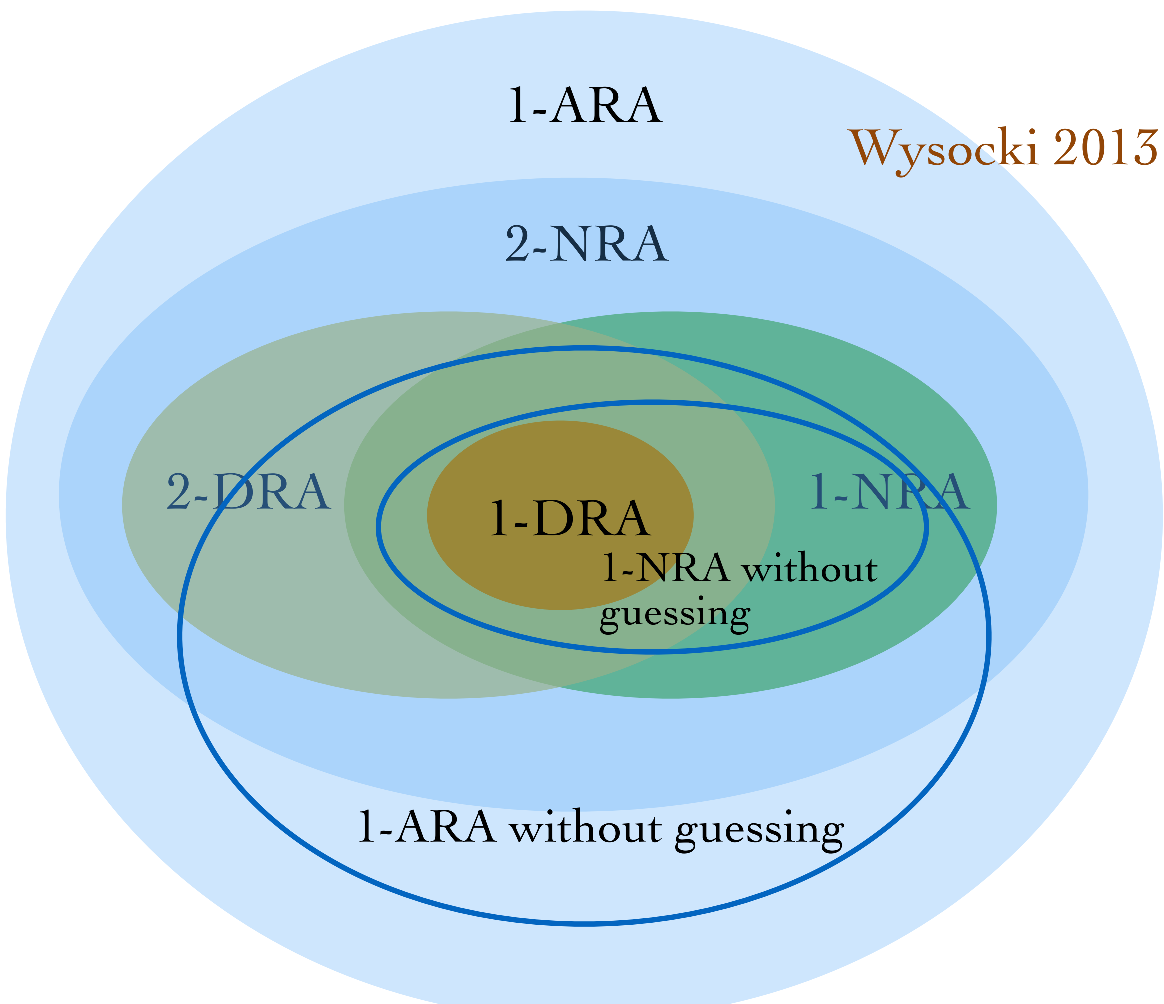
Non-robustness of register automata

1-way or 2-way { • ARA • NRA • URA } with guessing or not

• DRA = left congruence of orbit-finite index ⊃ orbit-finite monoids

Bojańczyk, Klin, L. 2014

rational expressions?



Rational (regular) expressions

- **unification based** rational expressions Kaminski, Tan 2006 only =
- rational expressions **with binders** Kurz, Suzuki, Tuosto 2012
- rational expressions **with memory** Libkin, Tan, Vrgoc 2015 φ
- rational expressions **with side-effects** Bojańczyk 201?
- rational **bar-**expressions Schroeder, Kozen, Milius, Wissmann 2017
- rational expressions **with (de)allocation** Brunet, Silva 2019
- **orbit-finite** rational expressions too weak

Rational (regular) expressions

- **unification based** rational expressions Kaminski, Tan 2006 only =
- rational expressions **with binders** Kurz, Suzuki, Tuosto 2012
- rational expressions **with memory** Libkin, Tan, Vrgoc 2015 φ
- rational expressions **with side-effects** Bojańczyk 201?
- rational **bar-**expressions Schroeder, Kozen, Milius, Wissmann 2017
- rational expressions **with (de)allocation** Brunet, Silva 2019
- **orbit-finite** rational expressions too weak

Question: Is there a notion of rational expressions in the monoid $\mathbb{A}^*$ of data words?

Rational (regular) expressions

- **unification based** rational expressions Kaminski, Tan 2006 only =
- rational expressions **with binders** Kurz, Suzuki, Tuosto 2012
- rational expressions **with memory** Libkin, Tan, Vrgoc 2015 φ
- rational expressions **with side-effects** Bojańczyk 201?
- rational **bar-**expressions Schroeder, Kozen, Milius, Wissmann 2017
- rational expressions **with (de)allocation** Brunet, Silva 2019
- **orbit-finite** rational expressions too weak

Question: Is there a notion of rational expressions in the monoid $\mathbb{A}^*$ of data words?

Question (Klin)**:** Is there an orbit-finite set of constructions that generates all NRA languages?

Rational (regular) expressions

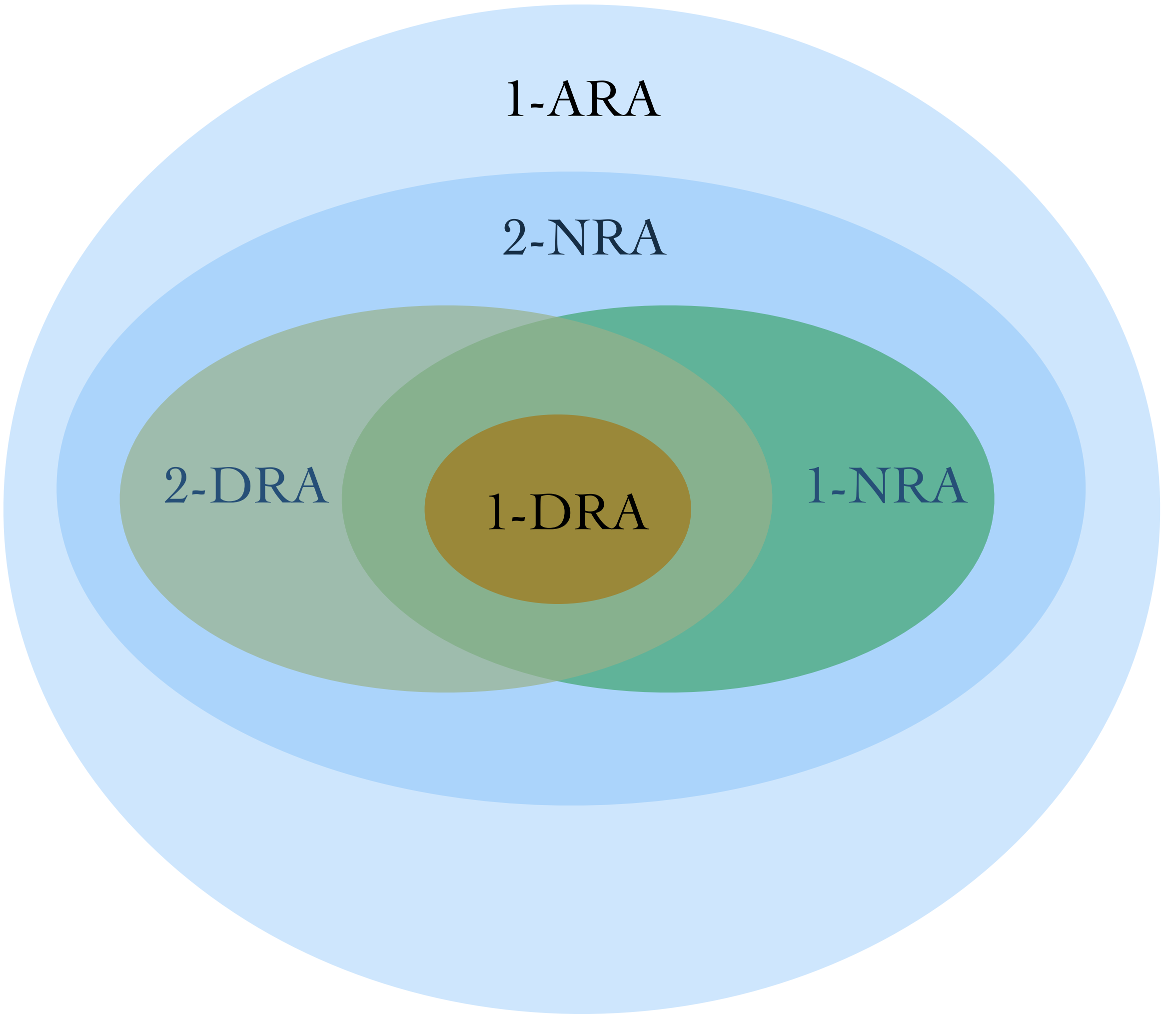
- **unification based** rational expressions — Kaminski, Tan 2006 — only =
- rational expressions **with binders** — Kurz, Suzuki, Tuosto 2012
- rational expressions **with memory** — Libkin, Tan, Vrgoc 2015 — φ
- rational expressions **with side-effects** — Bojańczyk 201?
- rational **bar-**expressions — Schroeder, Kozen, Milius, Wissmann 2017
- rational expressions **with (de)allocation** — Brunet, Silva 2019
- **orbit-finite** rational expressions — too weak

Question: Is there a notion of rational expressions in the monoid $\mathbb{A}^*$ of data words?

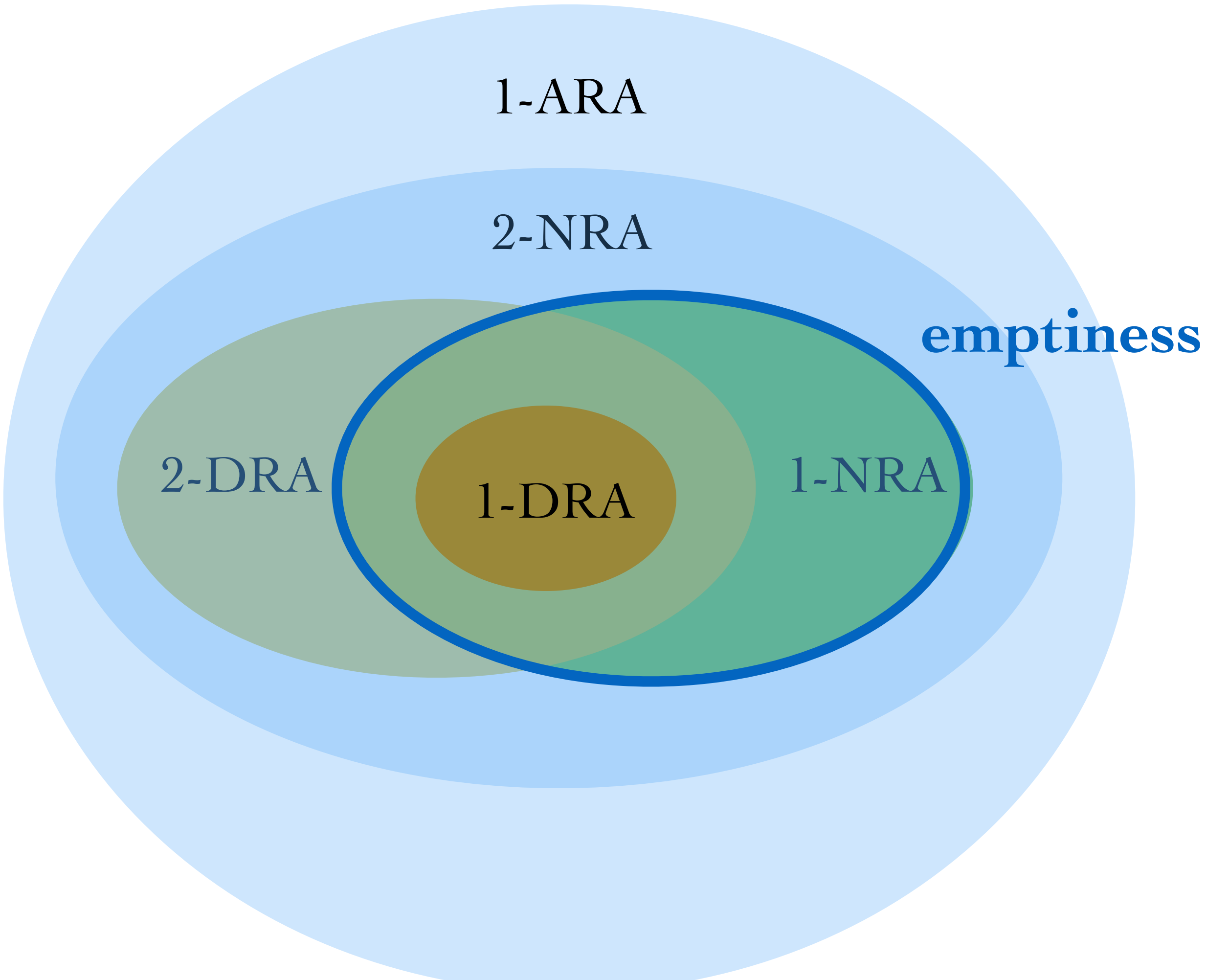
Question (Klin): Is there an orbit-finite set of constructions that generates all NRA languages?

Question (Klin): What should the question be?

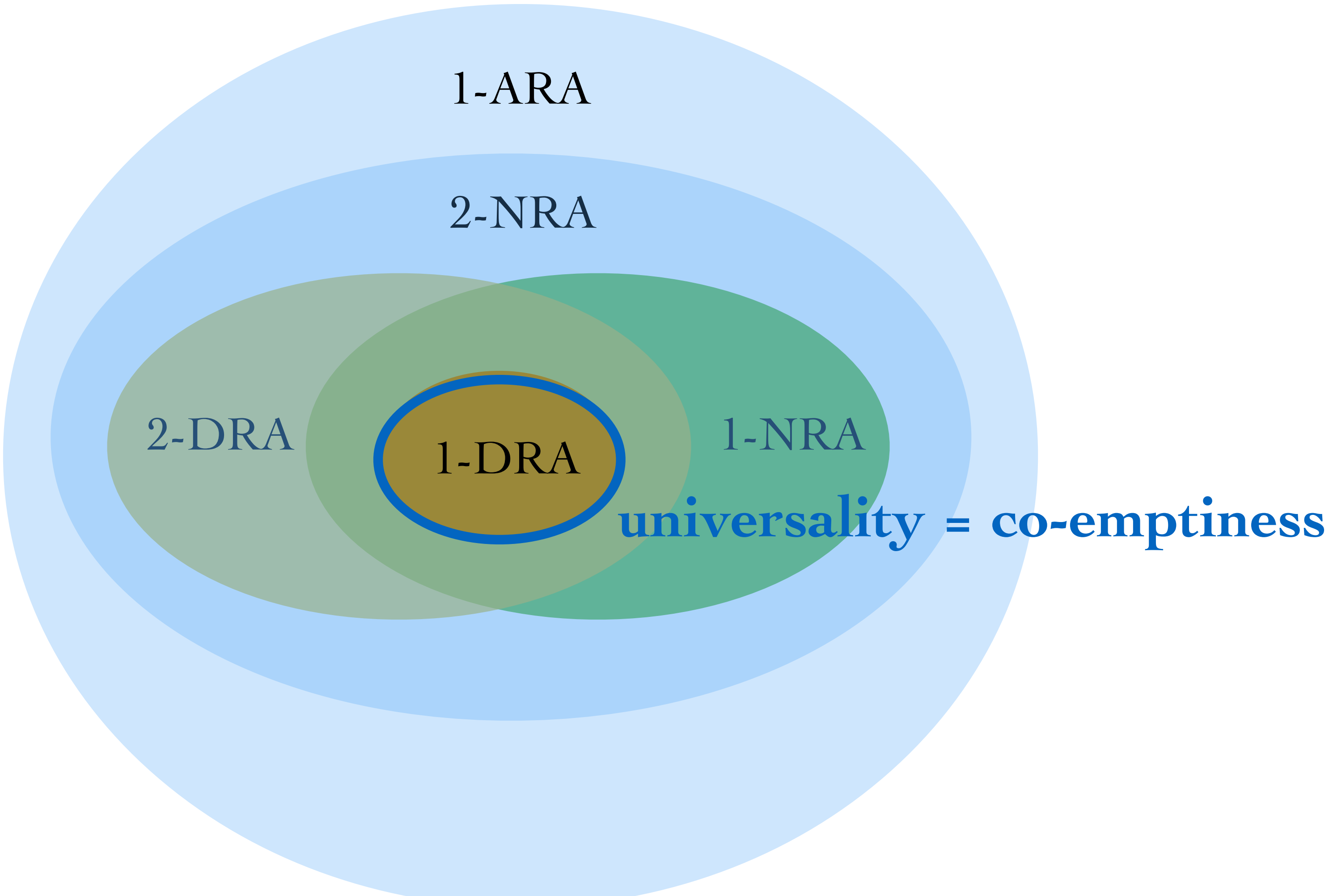
Decidability



Decidability



Decidability



II. Some recent advances

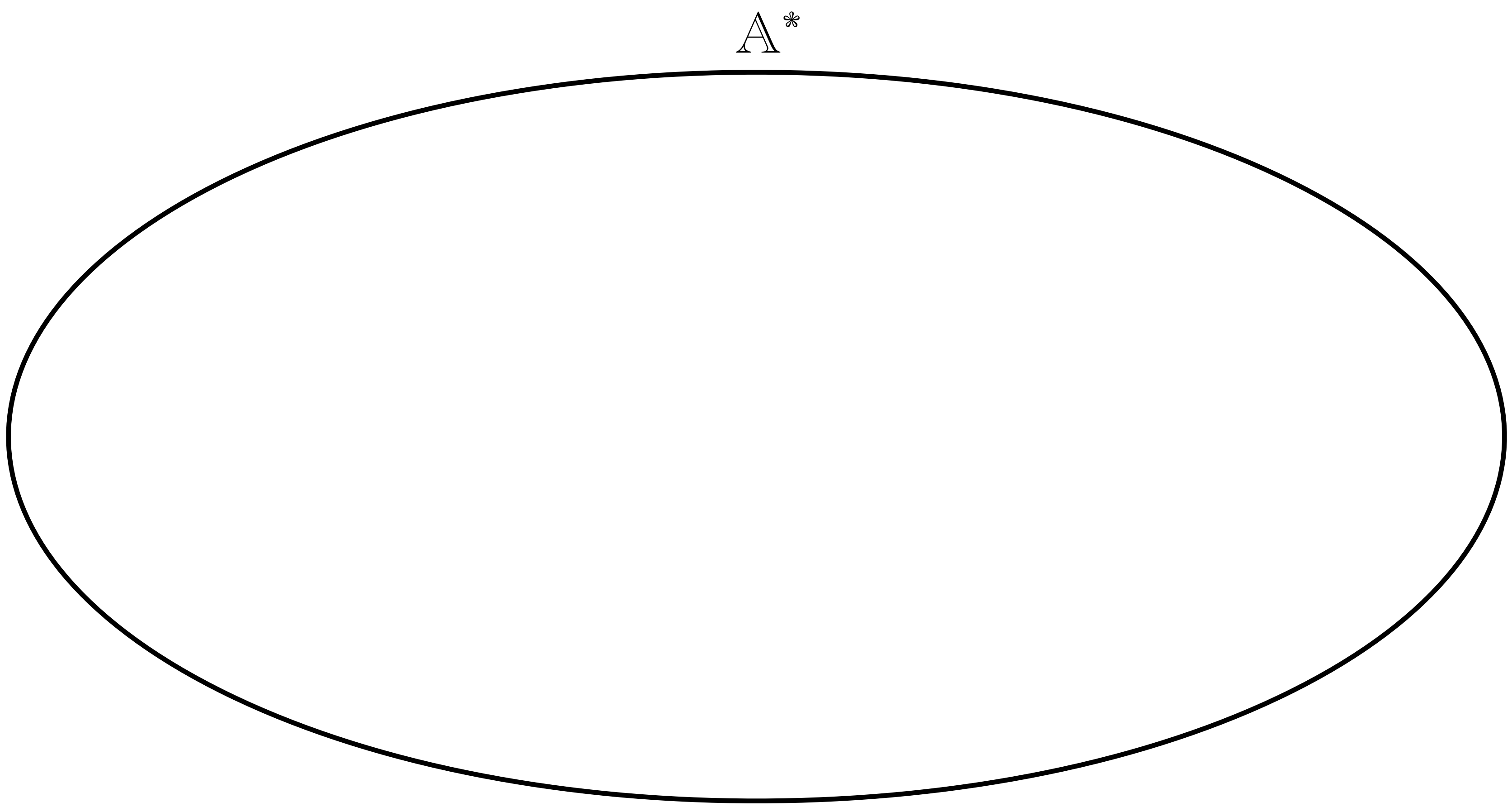
1) Deterministic separability
2) Deterministic collapse
3) Commutative images
4) Single-use registers

1) Deterministic separability

Motivation: DRA are decidable while NRA are not

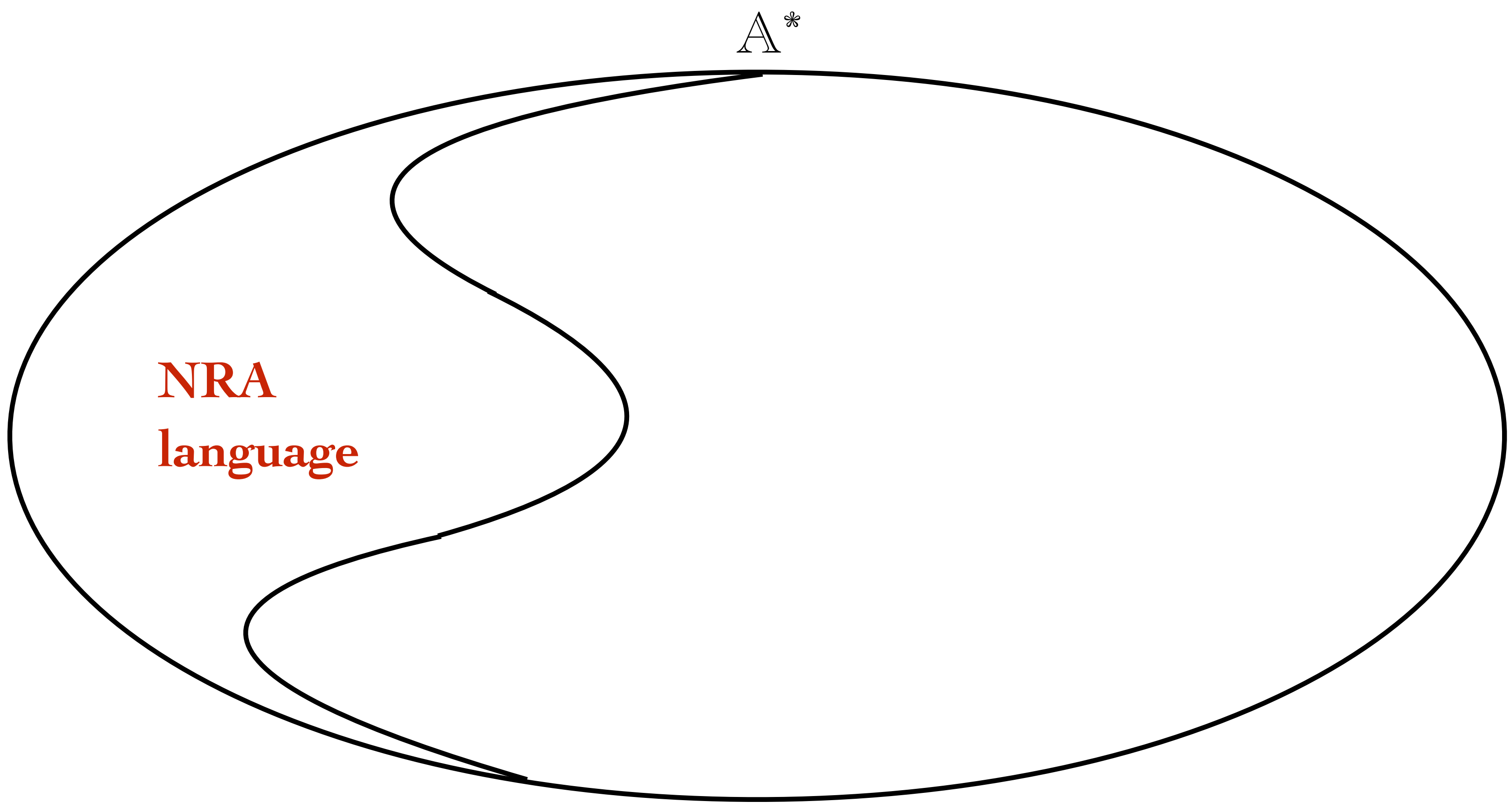
1) Deterministic separability

Motivation: DRA are decidable while NRA are not



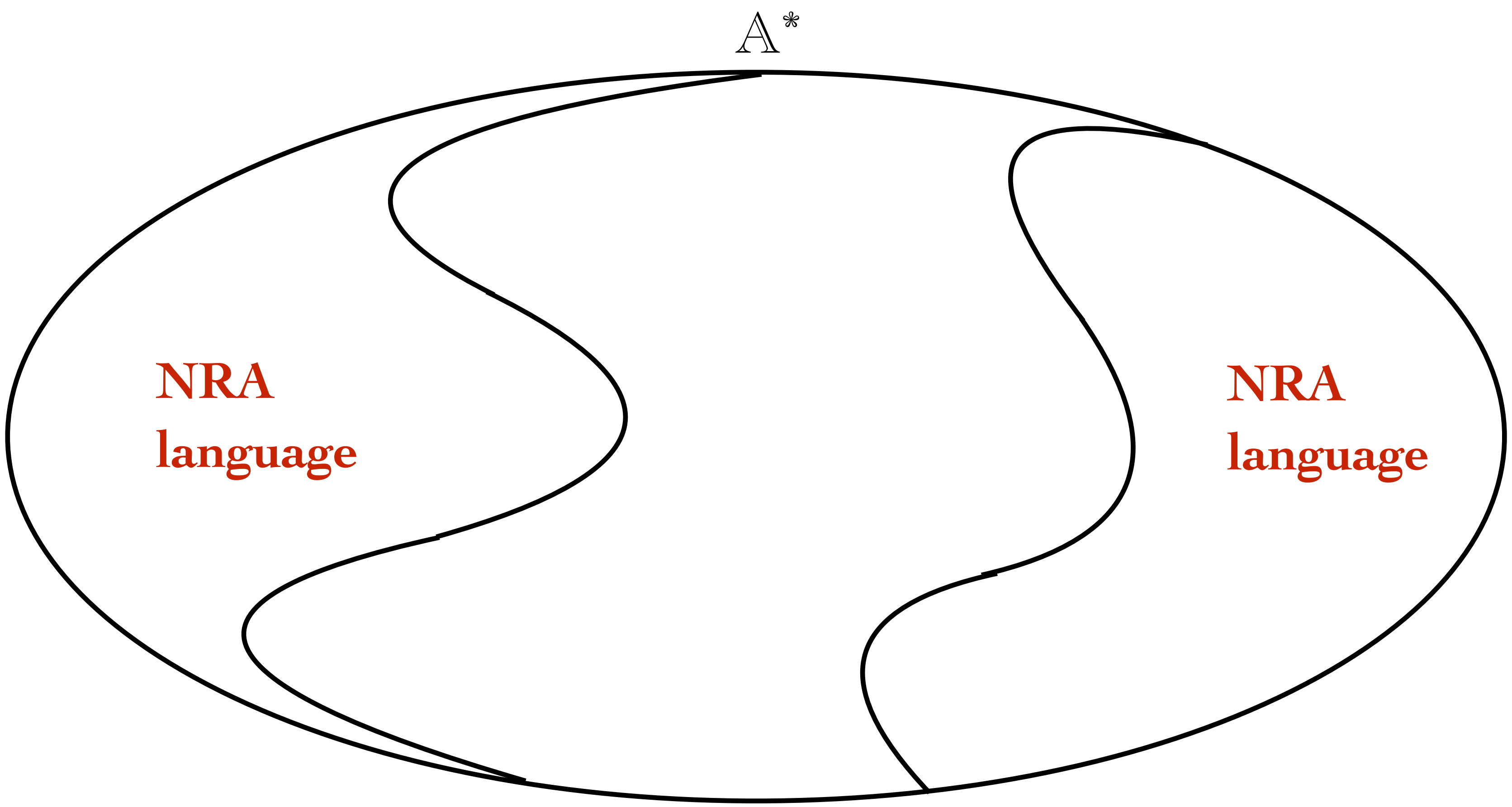
1) Deterministic separability

Motivation: DRA are decidable while NRA are not



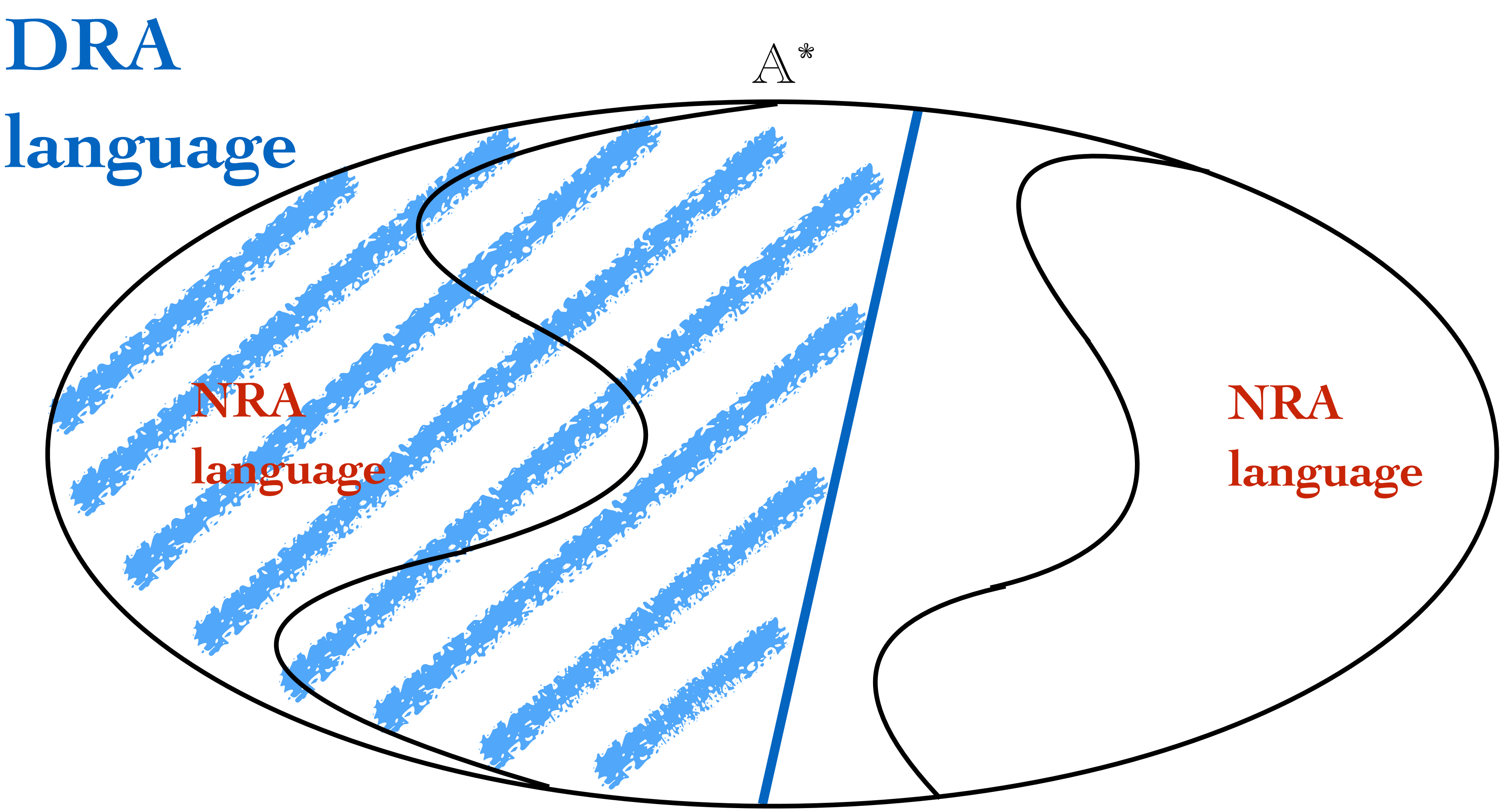
1) Deterministic separability

Motivation: DRA are decidable while NRA are not



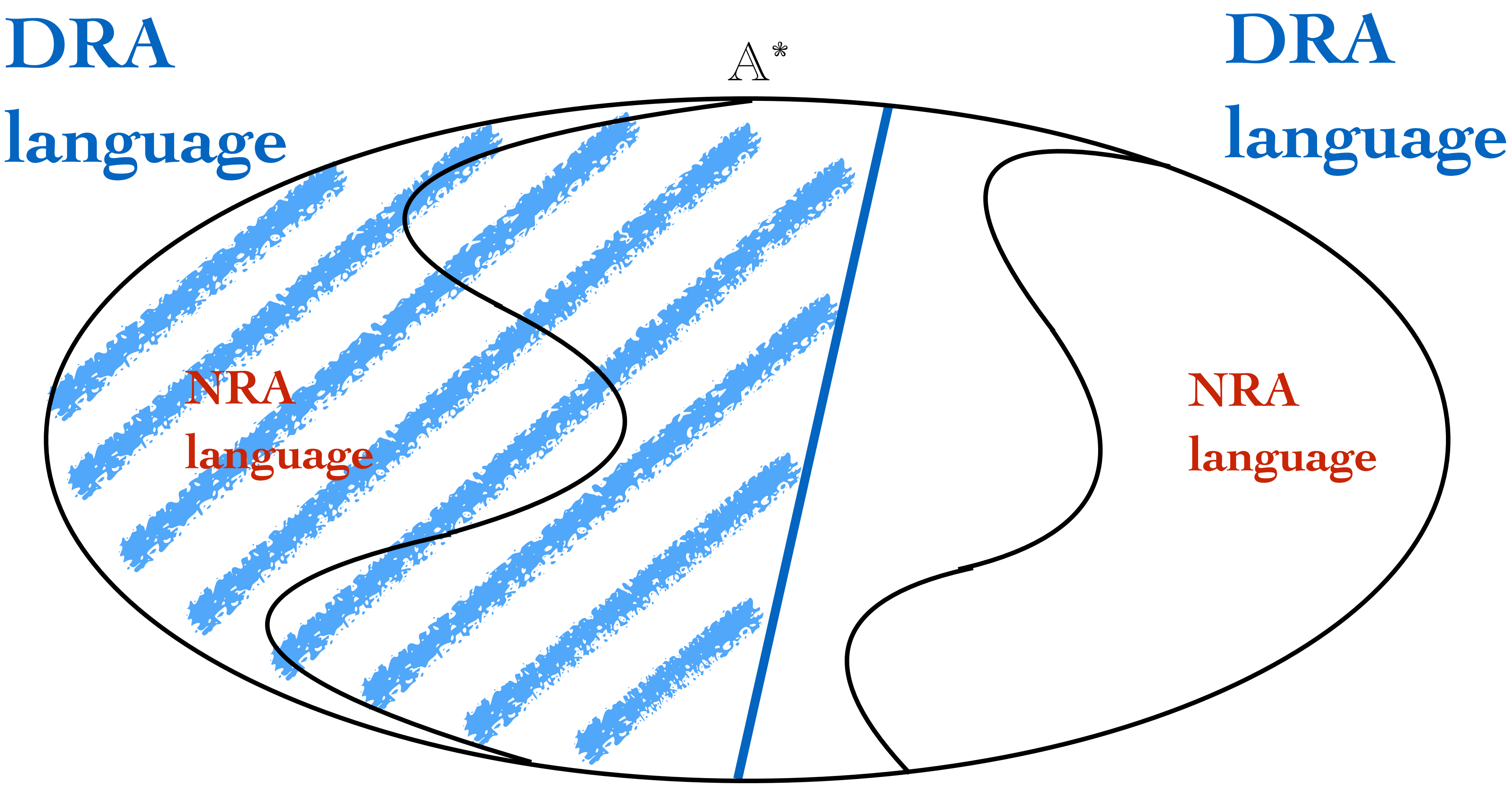
1) Deterministic separability

Motivation: DRA are decidable while NRA are not



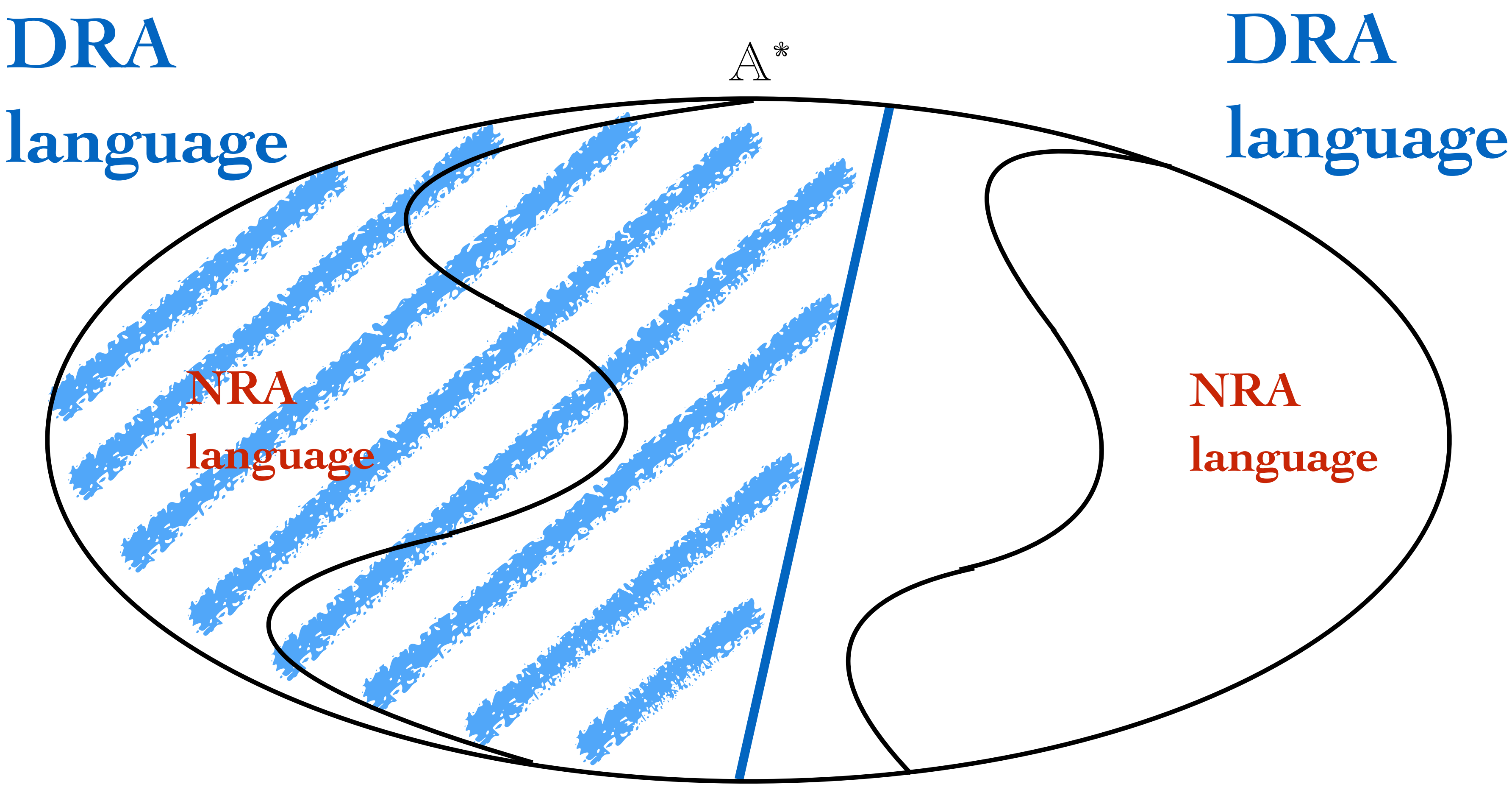
1) Deterministic separability

Motivation: DRA are decidable while NRA are not



1) Deterministic separability

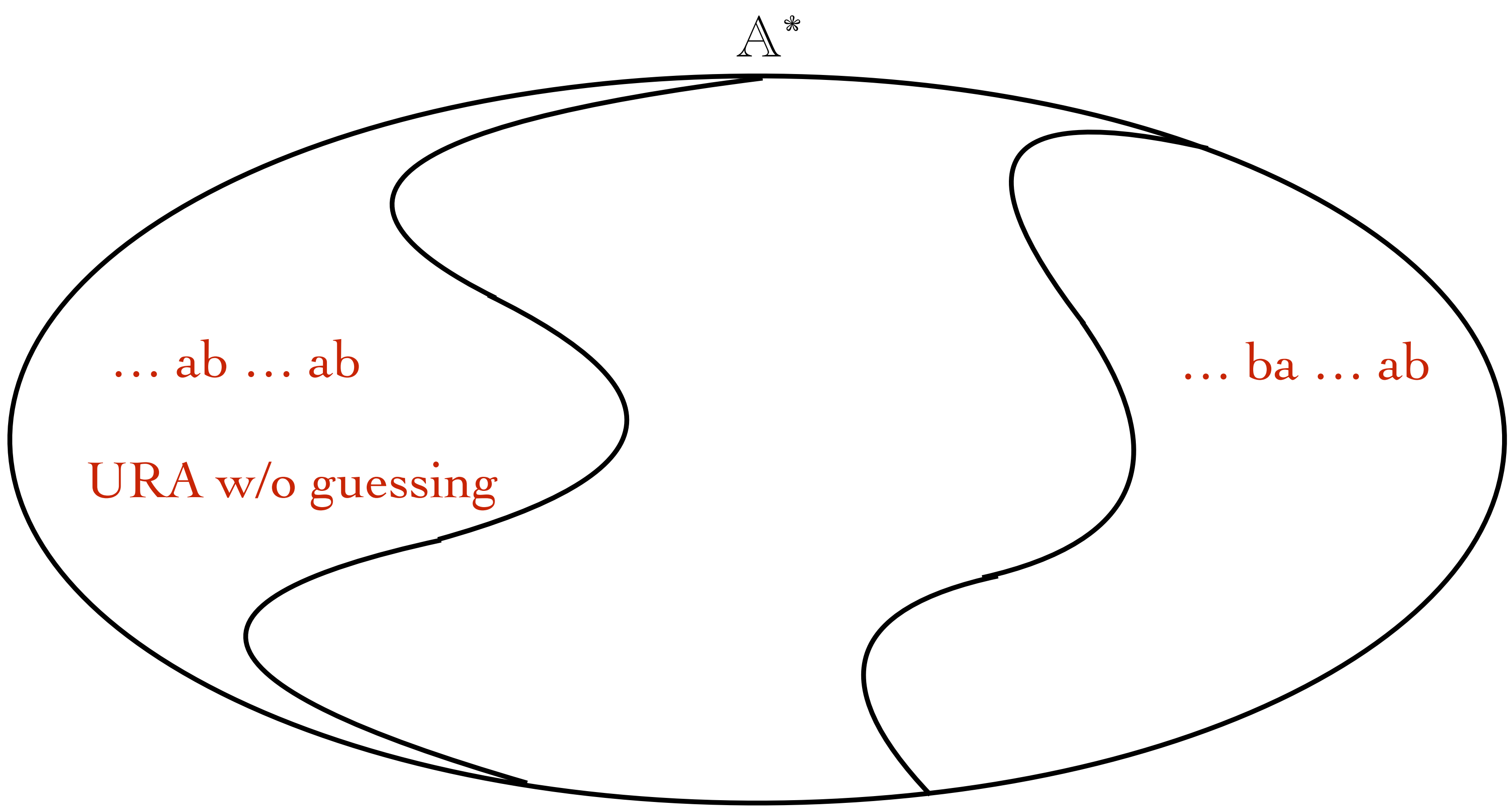
Motivation: DRA are decidable while NRA are not



deterministic separator = classifier

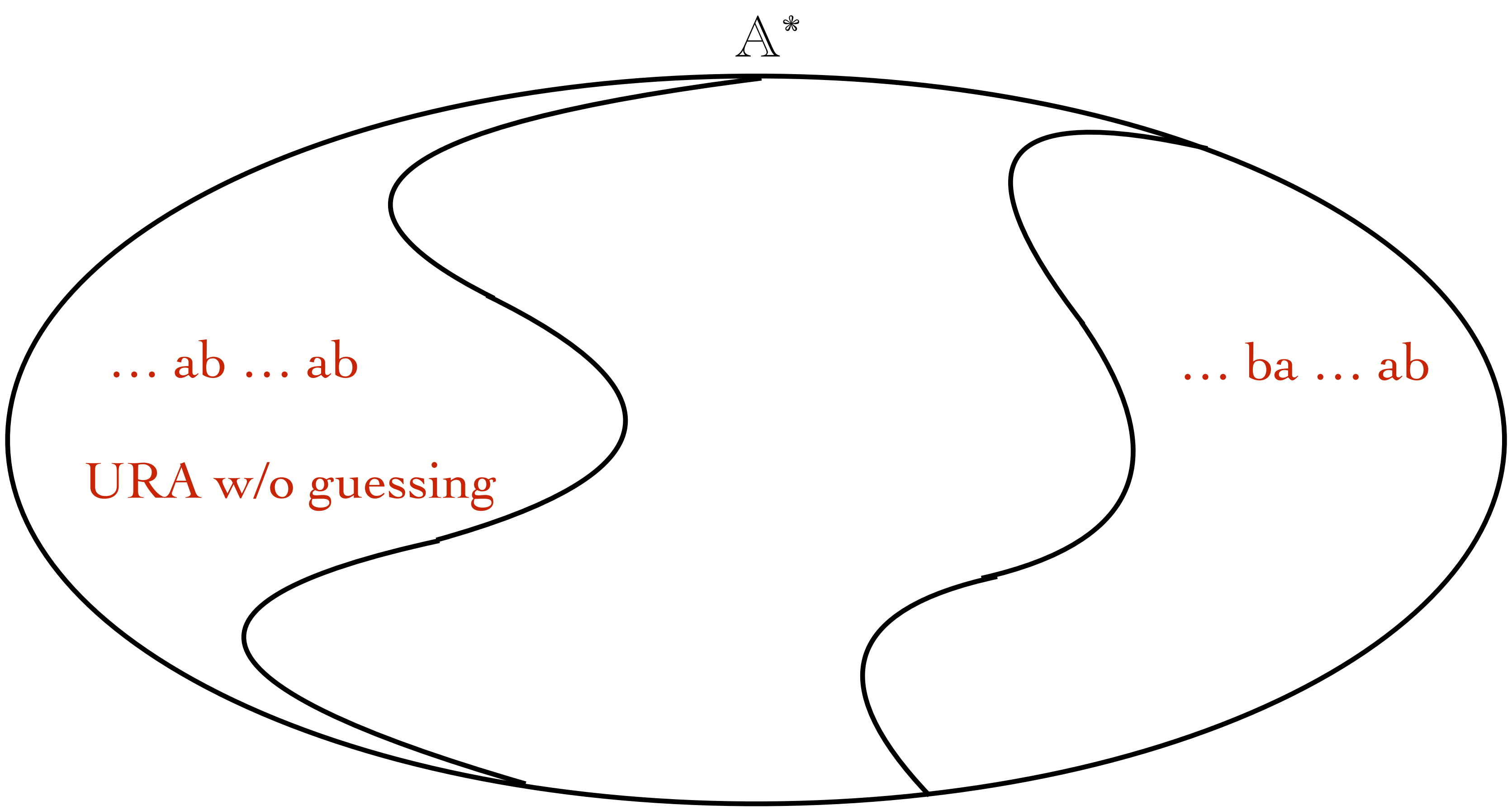
1) Deterministic separability

Example: no deterministic separator



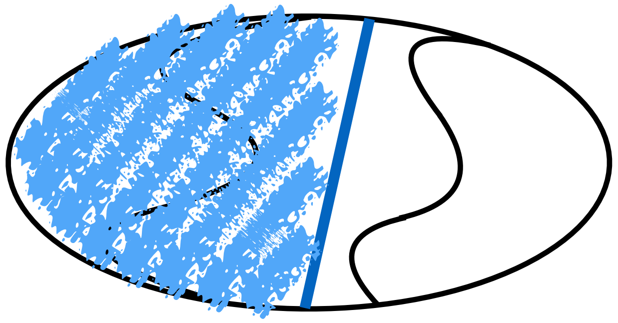
1) Deterministic separability

Example: no deterministic separator



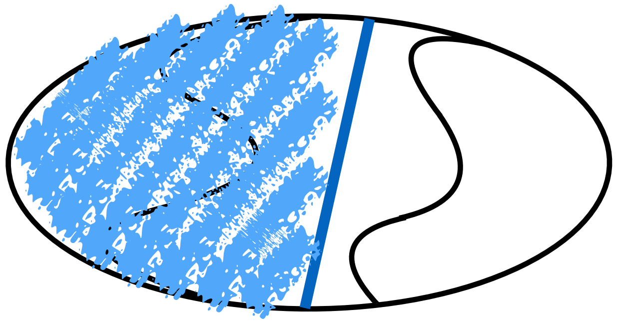
Conjecture (Colcombet 2012): there is always an **unambiguous** separator

1) Deterministic separability



Question: Is DRA-separability decidable for NRA?

1) Deterministic separability

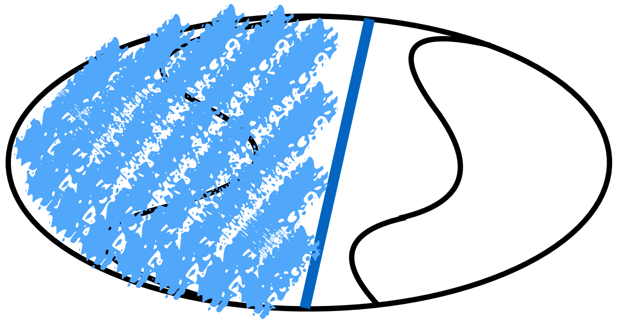


Question: Is DRA-separability decidable for NRA?

Partial answer (Clemente, L., Piórkowski 2020**):**
For every fixed *k*, *k*-DRA-separability is decidable for NRA.

timed
automata

1) Deterministic separability



Question: Is DRA-separability decidable for NRA?

Partial answer (Clemente, L., Piórkowski 2020):
For every fixed *k*, *k*-DRA-separability is decidable for NRA.

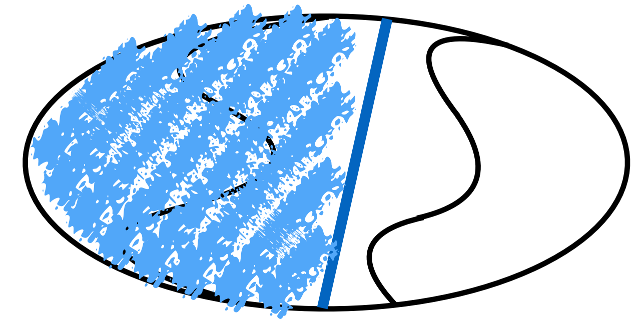
timed
automata

Key tool: Buchi-Landweber games over data words.

Alice:

Bob:

1) Deterministic separability



Question: Is DRA-separability decidable for NRA?

Partial answer (Clemente, L., Piórkowski 2020):
For every fixed k, k-DRA-separability is decidable for NRA.

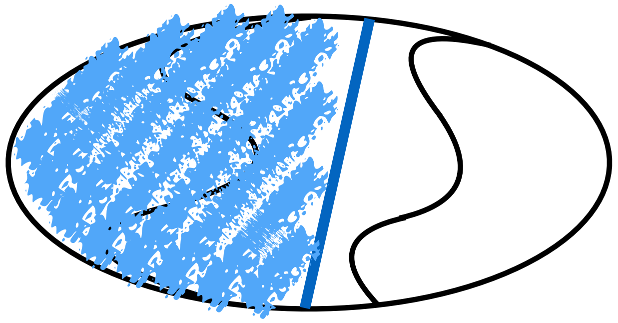
timed
automata

Key tool: Buchi-Landweber games over data words.

Alice: a_1

Bob:

1) Deterministic separability



Question: Is DRA-separability decidable for NRA?

timed automata

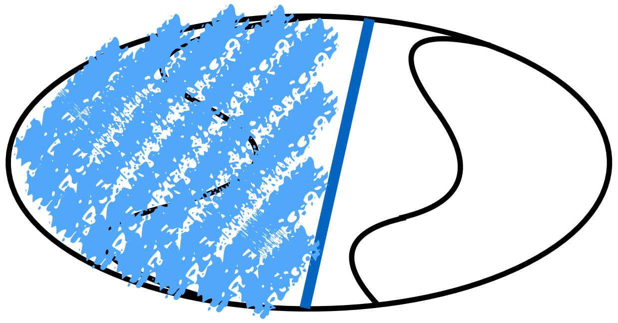
Partial answer (Clemente, L., Piórkowski 2020):
For every fixed *k*, *k*-DRA-separability is decidable for NRA.

Key tool: Buchi-Landweber games over data words.

Alice: a_1

Bob: b_1

1) Deterministic separability



Question: Is DRA-separability decidable for NRA?

Partial answer (Clemente, L., Piórkowski 2020)**:**
For every fixed *k*, *k*-DRA-separability is decidable for NRA.

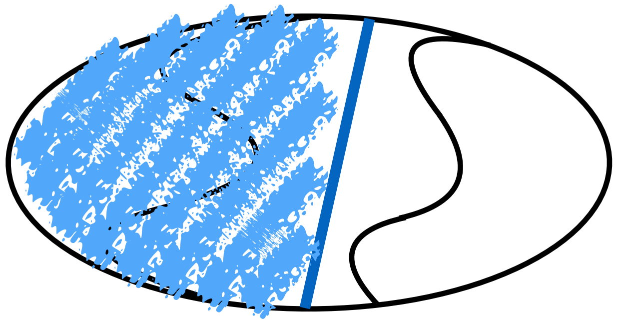
timed
automata

Key tool: Buchi-Landweber games over data words.

Alice: a_1 a_2

Bob: b_1

1) Deterministic separability



Question: Is DRA-separability decidable for NRA?

timed automata

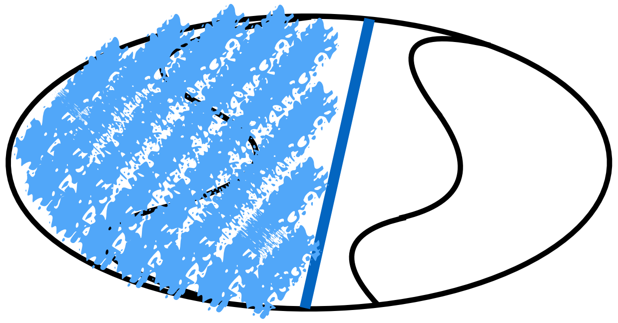
Partial answer (Clemente, L., Piórkowski 2020):
For every fixed k, k-DRA-separability is decidable for NRA.

Key tool: Buchi-Landweber games over data words.

Alice: a_1 a_2

Bob: b_1 b_2

1) Deterministic separability



Question: Is DRA-separability decidable for NRA?

Partial answer (Clemente, L., Piórkowski 2020)**:**
For every fixed k, k-DRA-separability is decidable for NRA.

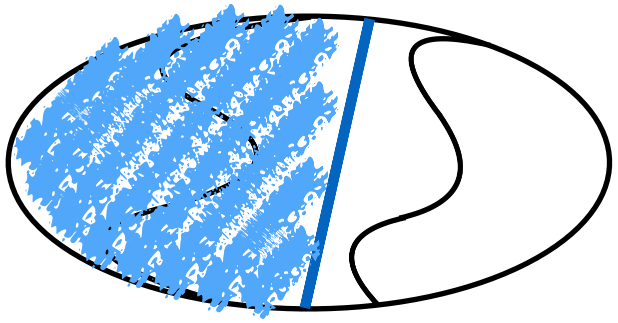
timed
automata

Key tool: Buchi-Landweber games over data words.

Alice: a_1 a_2 a_3

Bob: b_1 b_2

1) Deterministic separability



Question: Is DRA-separability decidable for NRA?

Partial answer (Clemente, L., Piórkowski 2020)**:**
For every fixed *k*, *k*-DRA-separability is decidable for NRA.

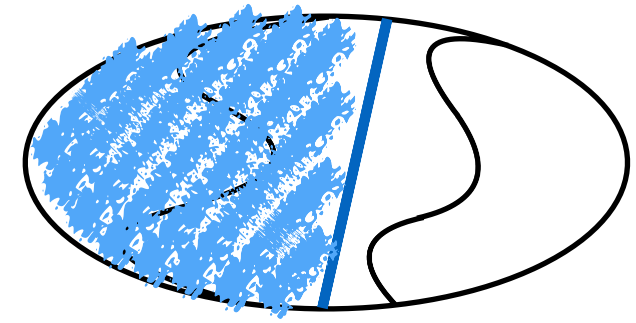
timed
automata

Key tool: Buchi-Landweber games over data words.

Alice: a_1 a_2 a_3

Bob: b_1 b_2 b_3

1) Deterministic separability



Question: Is DRA-separability decidable for NRA?

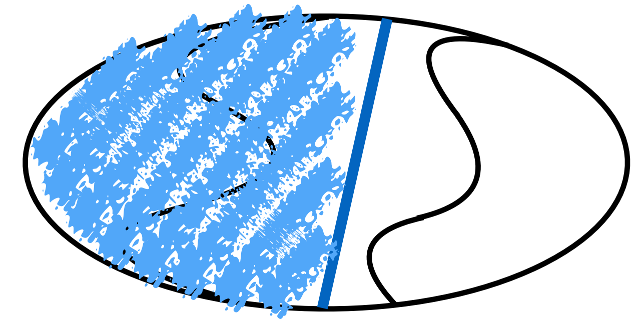
Partial answer (Clemente, L., Piórkowski 2020)**:**
For every fixed *k*, *k*-DRA-separability is decidable for NRA.

timed
automata

Key tool: Buchi-Landweber games over data words.

Alice: a_1 a_2 a_3 …
Bob: b_1 b_2 b_3

1) Deterministic separability



Question: Is DRA-separability decidable for NRA?

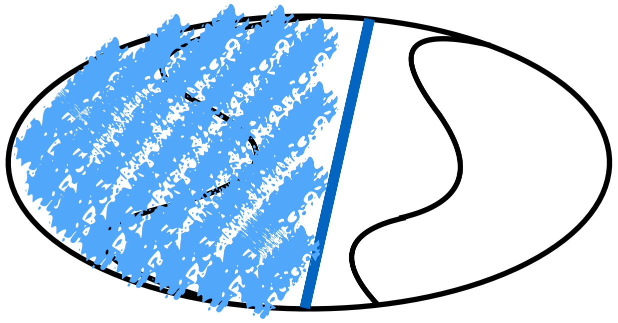
Partial answer (Clemente, L., Piórkowski 2020):
For every fixed *k*, *k*-DRA-separability is decidable for NRA.

timed
automata

Key tool: Buchi-Landweber games over data words.

Alice:	a_1	a_2	a_3	...	$\in \mathbb{A}^\omega$
Bob:	b_1	b_2	b_3		$\in \mathbb{B}^\omega$

1) Deterministic separability



Question: Is DRA-separability decidable for NRA?

Partial answer (Clemente, L., Piórkowski 2020):
For every fixed k, k-DRA-separability is decidable for NRA.

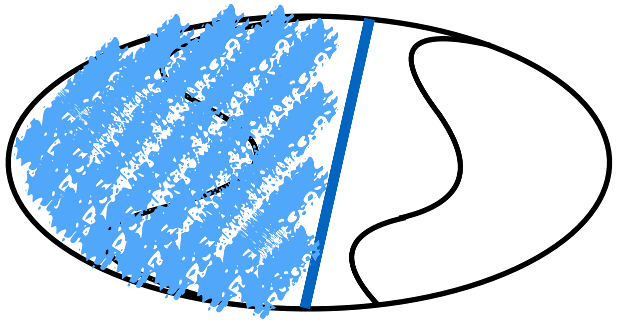
timed automata

Key tool: Buchi-Landweber games over data words.

Alice:	a_1	a_2	a_3	...	$\in \mathbb{A}^\omega$
Bob:	b_1	b_2	b_3		$\in B^\omega$

Winning condition (for *Alice*): $W \subseteq (\mathbb{A} \times B)^\omega$ recognized by NRA

1) Deterministic separability



Question: Is DRA-separability decidable for NRA?

Partial answer (Clemente, L., Piórkowski 2020):
For every fixed k, k-DRA-separability is decidable for NRA.

timed automata

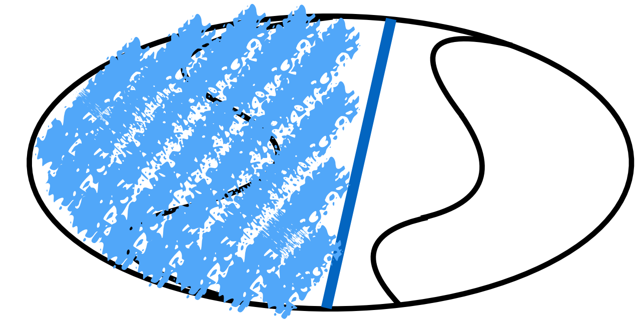
Key tool: Buchi-Landweber games over data words.

Alice:	a_1	a_2	a_3	...	$\in \mathbb{A}^\omega$
Bob:	b_1	b_2	b_3		$\in B^\omega$

Winning condition (for *Alice*): $W \subseteq (\mathbb{A} \times B)^\omega$ recognized by NRA

Theorem: For every fixed k, it is decidable if *Bob* has a k-register winning strategy (= k-DRA **with output B**).

1) Deterministic separability



Question: Is DRA-separability decidable for NRA?

Partial answer (Clemente, L., Piórkowski 2020)**:**
For every fixed *k*, *k*-DRA-separability is decidable for NRA.

timed
automata

Key tool: Buchi-Landweber games over data words.

Alice: $a_1 \quad a_2 \quad a_3 \quad \ldots \quad \in \mathbb{A}^\omega$

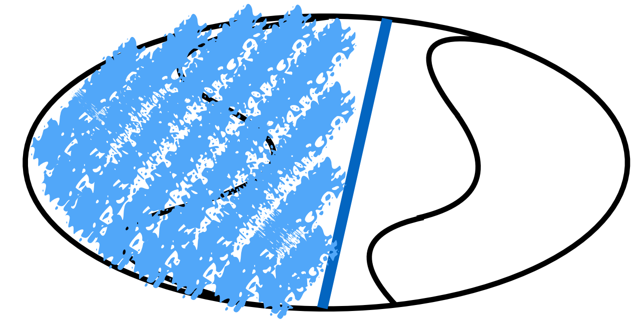
Bob: $b_1 \quad b_2 \quad b_3 \quad \ldots \quad \in B^\omega$

Winning condition (for *Alice*): $W \subseteq (\mathbb{A} \times B)^\omega$ recognized by NRA

Theorem: For every fixed *k*, it is decidable if *Bob* has a *k*-register winning strategy
(= *k*-DRA **with output B**).

Separability game: $B = \{accept, reject\}$

1) Deterministic separability



Question: Is DRA-separability decidable for NRA?

Partial answer (Clemente, L., Piórkowski 2020):
For every fixed *k*, *k*-DRA-separability is decidable for NRA.

timed automata

Key tool: Buchi-Landweber games over data words.

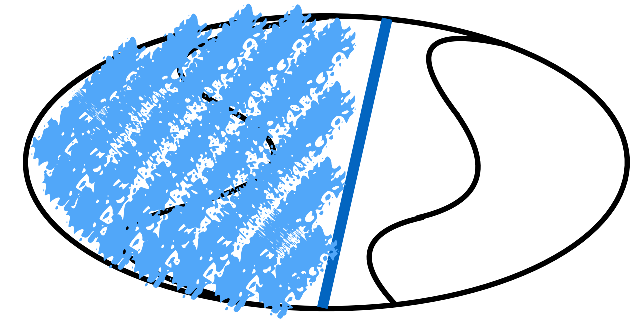
Alice:	a_1	a_2	a_3	…	$\in \mathbb{A}^\omega$
Bob:	b_1	b_2	b_3		$\in B^\omega$

Winning condition (for *Alice*): $W \subseteq (\mathbb{A} \times B)^\omega$ recognized by NRA

Theorem: For every fixed *k*, it is decidable if *Bob* has a *k*-register winning strategy (= *k*-DRA **with output B**).

Separability game: B = {*accept*, *reject*}
W = some prefix is incorrectly classified

1) Deterministic separability



Question: Is DRA-separability decidable for NRA?

Partial answer (Clemente, L., Piórkowski 2020):
For every fixed *k*, *k*-DRA-separability is decidable for NRA.

timed automata

Key tool: Buchi-Landweber games over data words.

Alice: $a_1 \quad a_2 \quad a_3 \quad \dots \quad \in \mathbb{A}^\omega$

Bob: $b_1 \quad b_2 \quad b_3 \quad \dots \quad \in B^\omega$

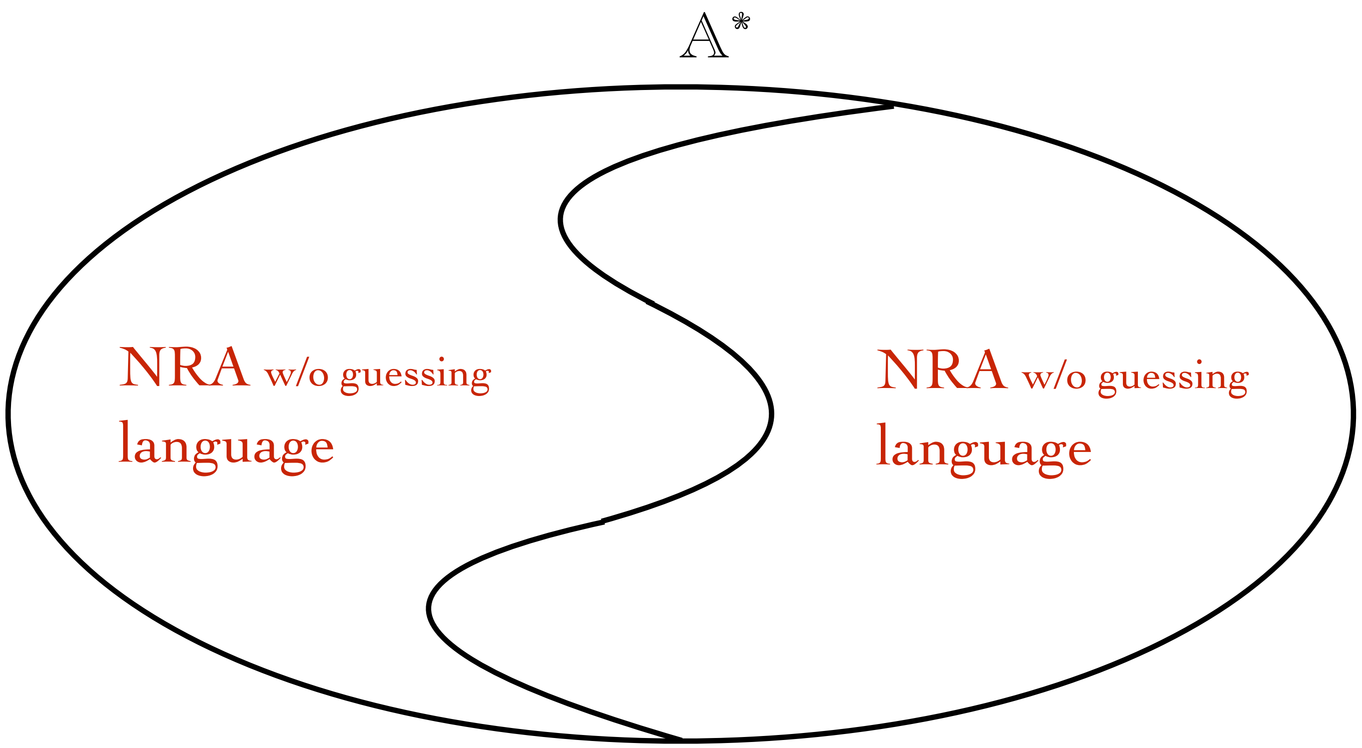
Winning condition (for *Alice*): $W \subseteq (\mathbb{A} \times B)^\omega$ recognized by NRA

Theorem: For every fixed *k*, it is decidable if *Bob* has a *k*-register winning strategy (= *k*-DRA **with output B**).

Separability game: B = {*accept*, *reject*}
W = some prefix is incorrectly classified
k-register winning strategy = *k*-DRA separator

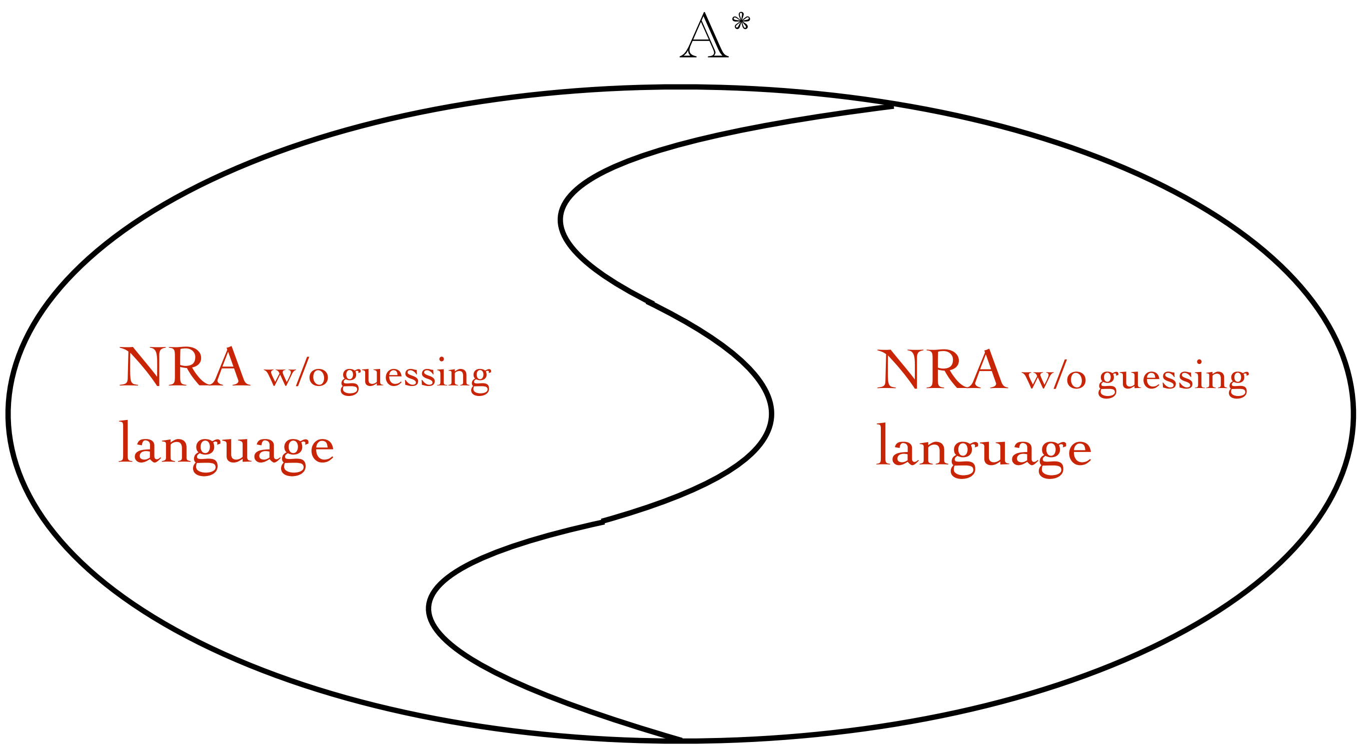
2) Deterministic collapse

Special case: two complementing DRA



2) Deterministic collapse

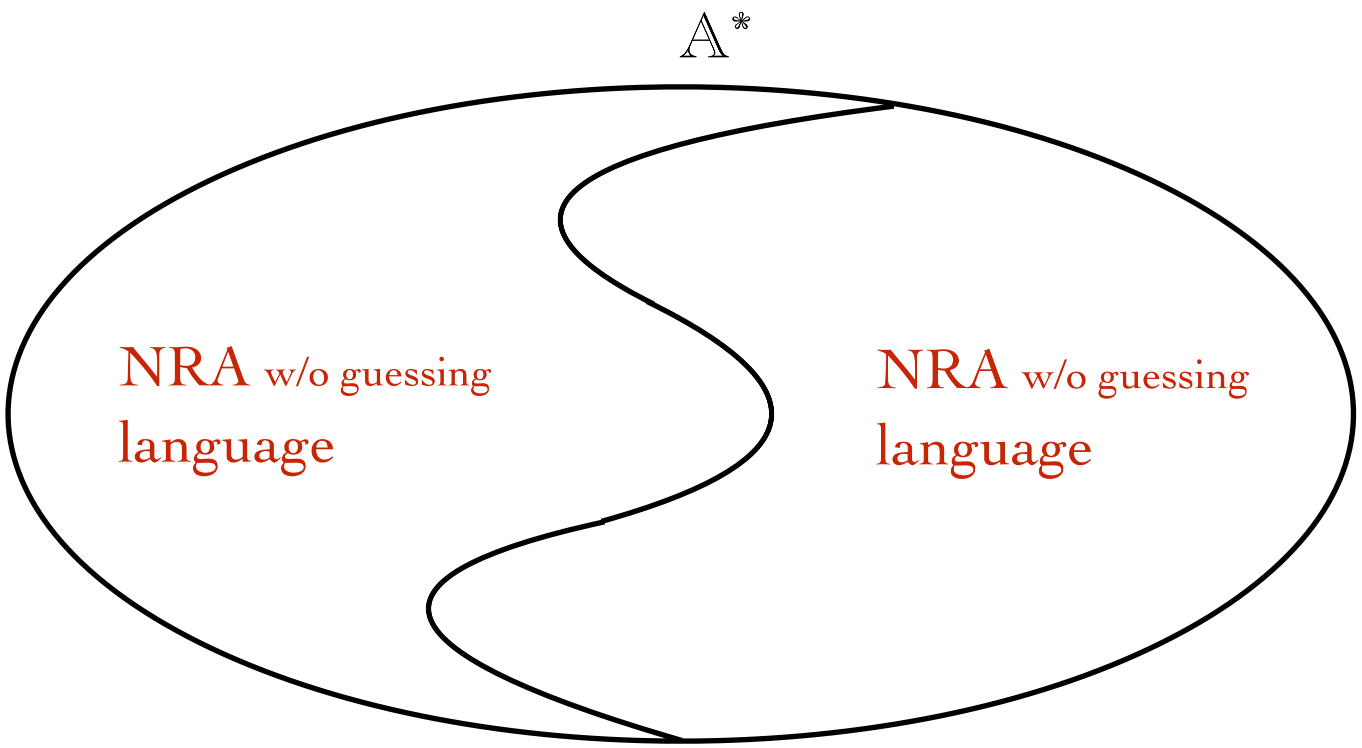
Special case: two complementing DRA



Theorem (Klin, L., Toruńczyk 2021): NRA co-NRA = DRA

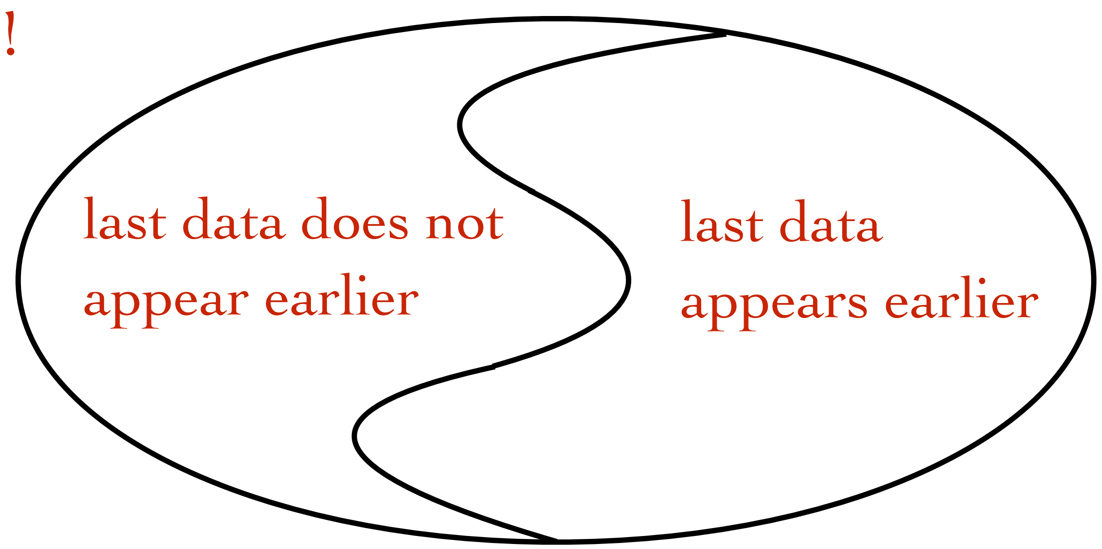
2) Deterministic collapse

Special case: two complementing DRA



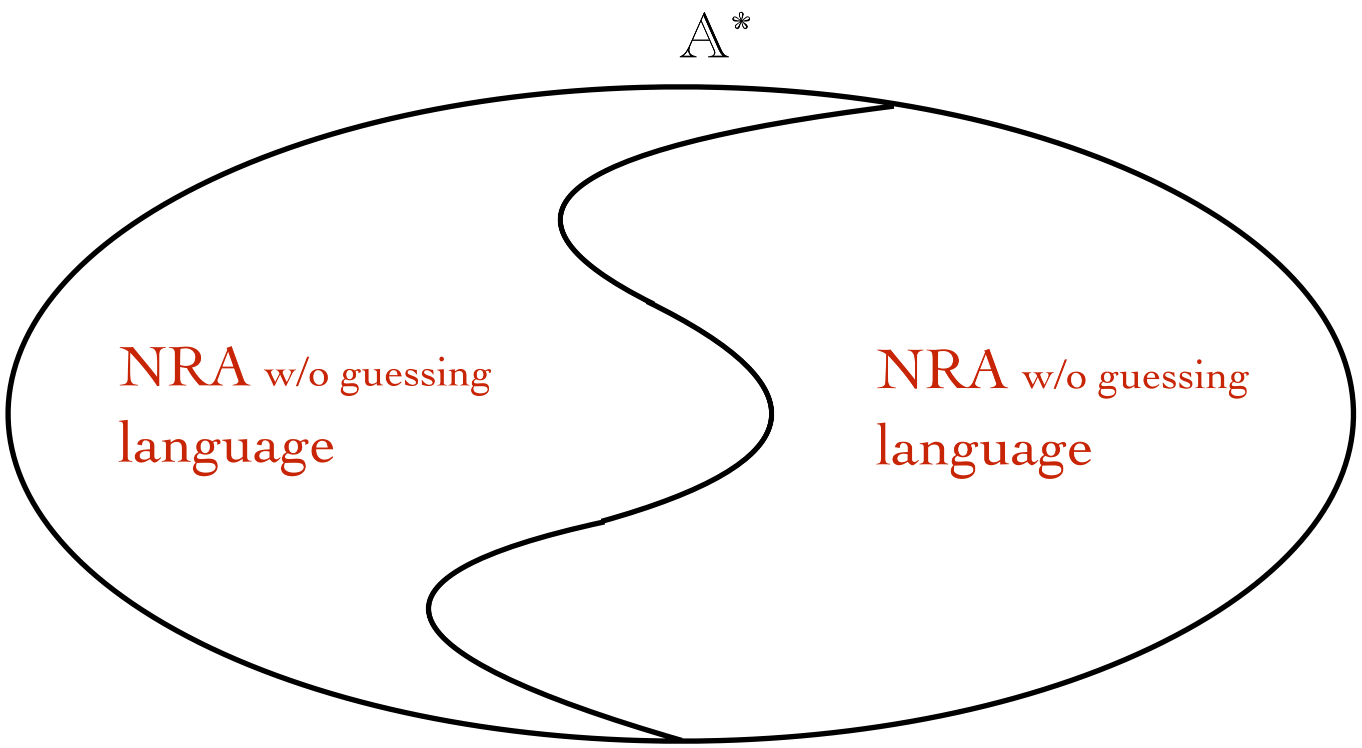
Theorem (Klin, L., Toruńczyk 2021): NRA $\cap$ co-NRA = DRA

Fails with guessing!



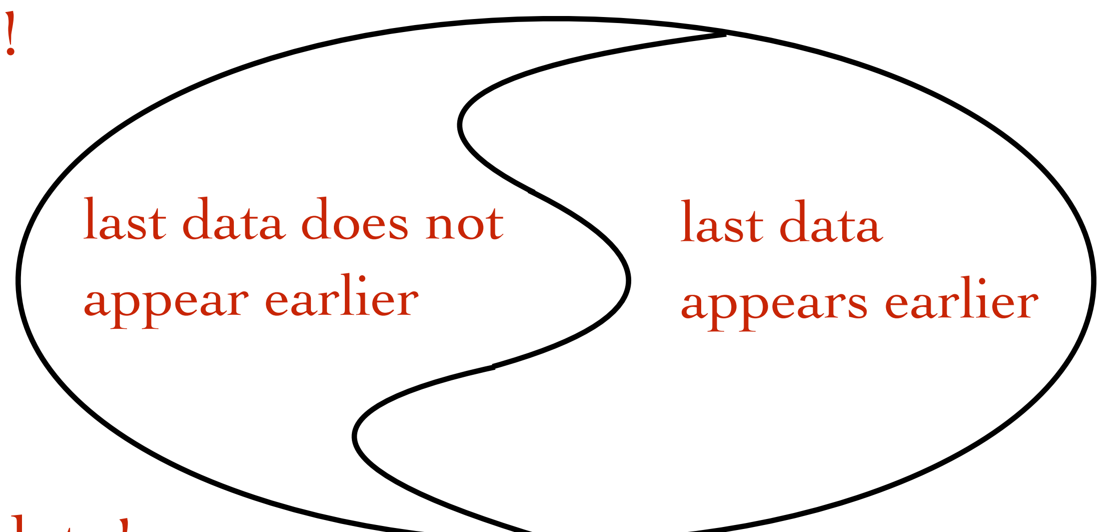
2) Deterministic collapse

Special case: two complementing DRA



Theorem (Klin, L., Toruńczyk 2021): NRA co-NRA = DRA

Fails with guessing!



Fails with ordered data!

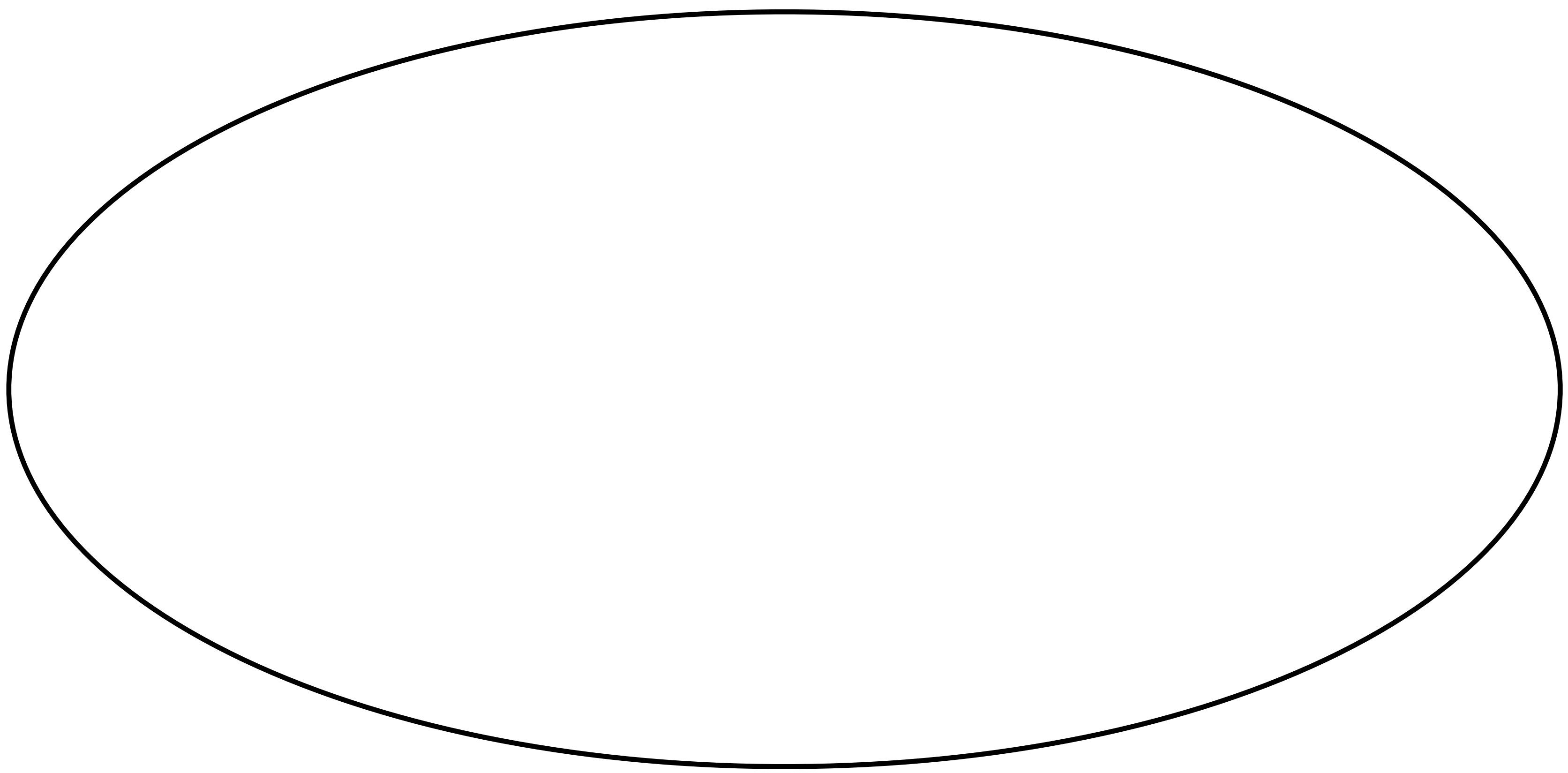
2) Deterministic collapse

Key tool: splitting lemma (+ orbit-finite Myhill-Nerode)

2) Deterministic collapse

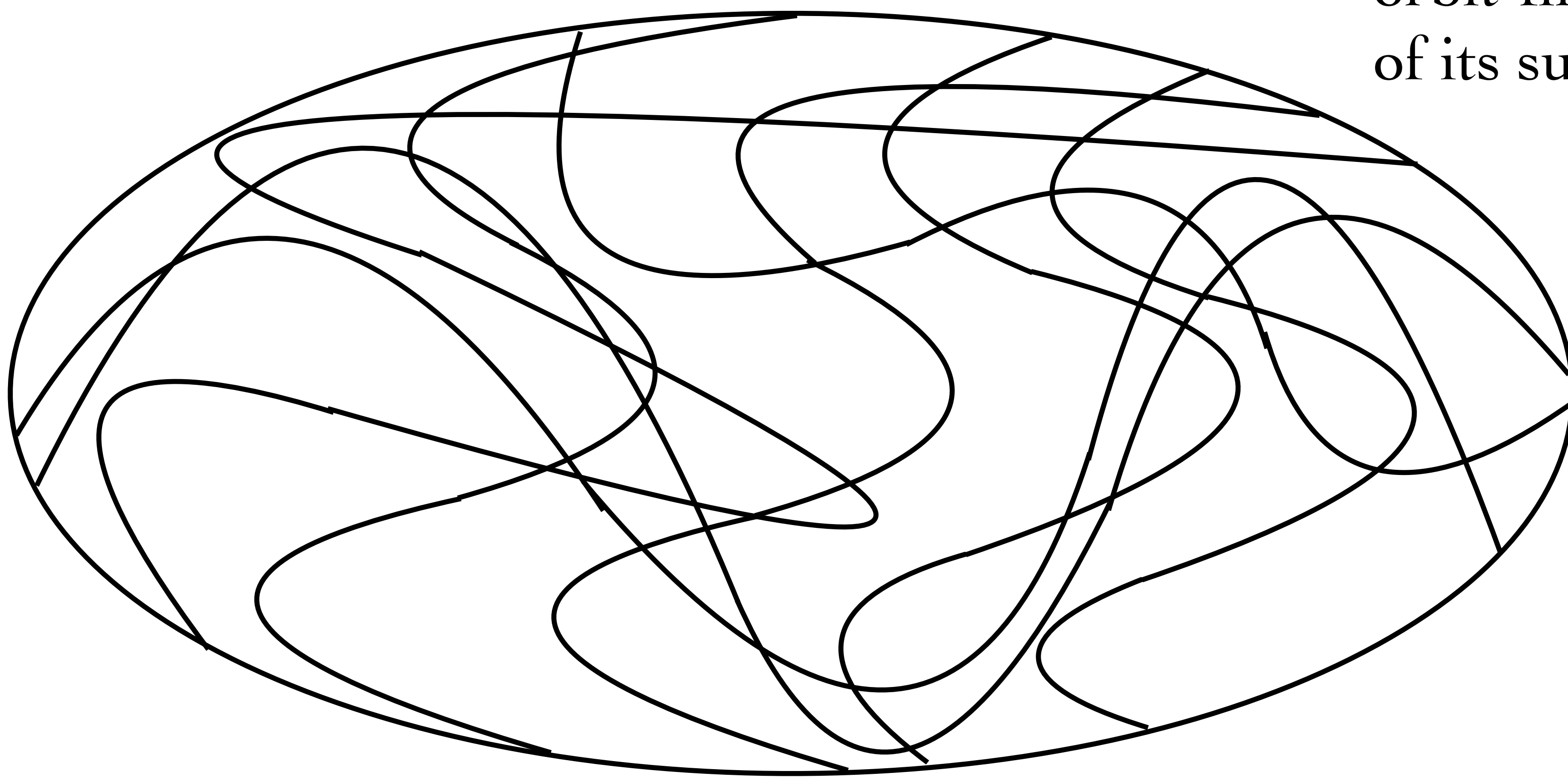
Key tool: splitting lemma (+ orbit-finite Myhill-Nerode)

an invariant set



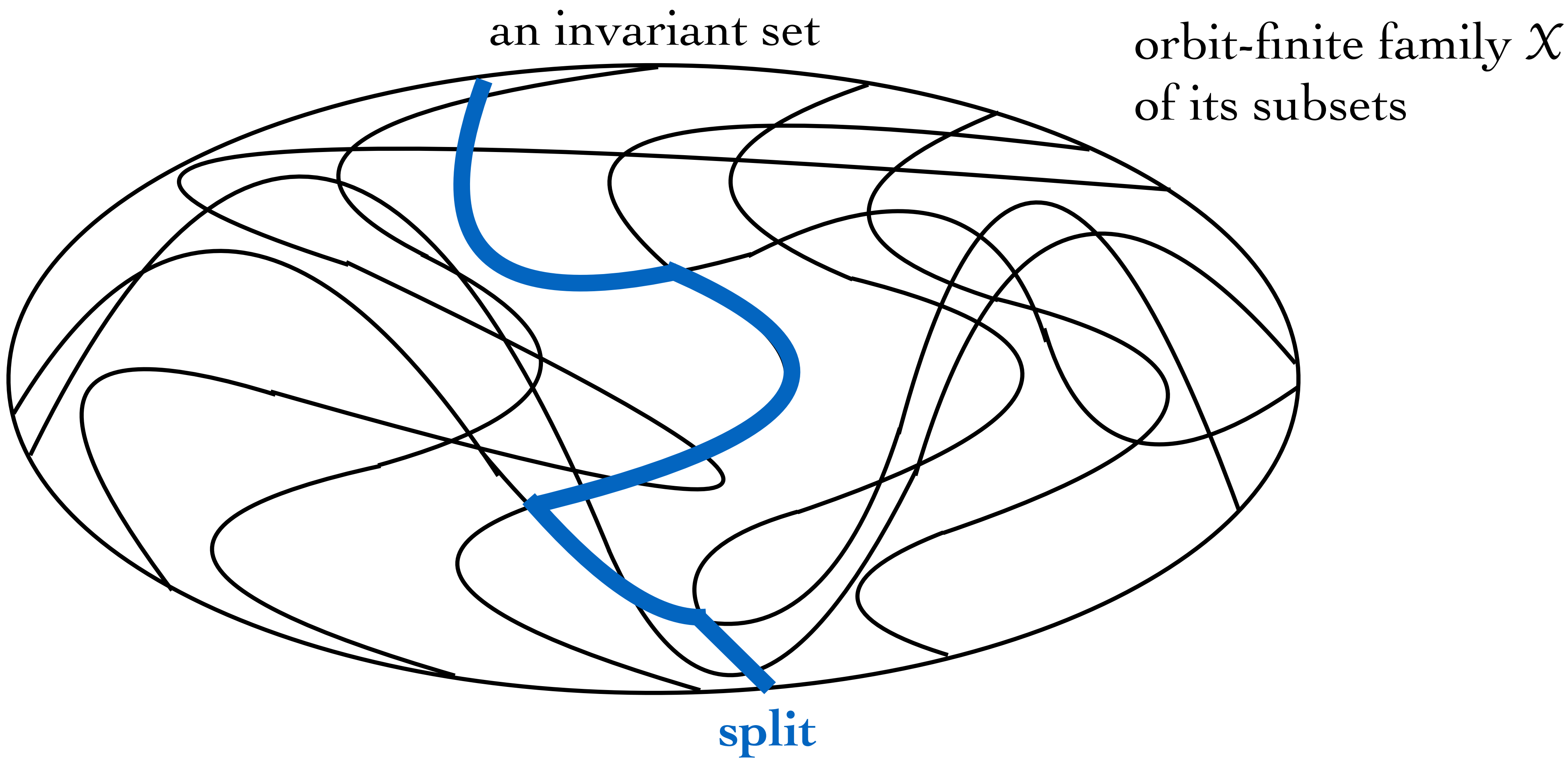
2) Deterministic collapse

Key tool: splitting lemma (+ orbit-finite Myhill-Nerode)



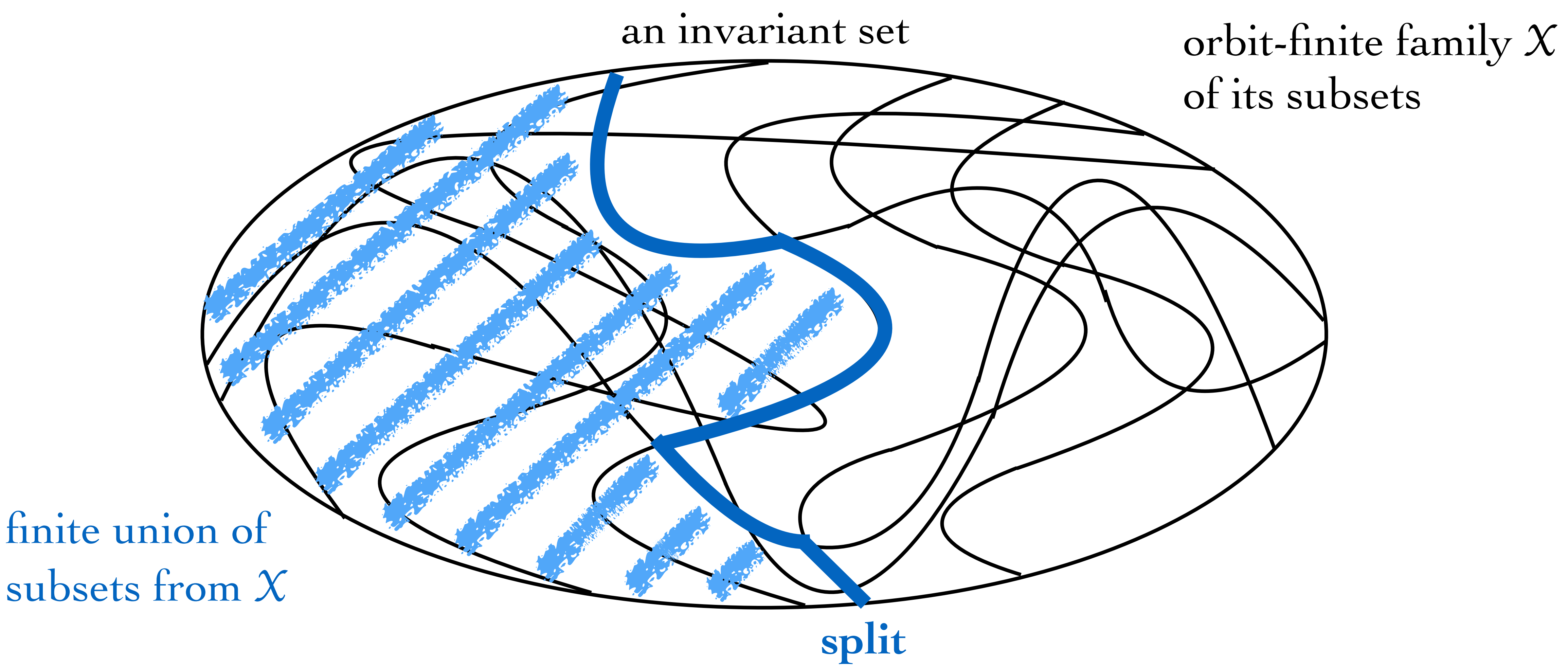
2) Deterministic collapse

Key tool: splitting lemma (+ orbit-finite Myhill-Nerode)



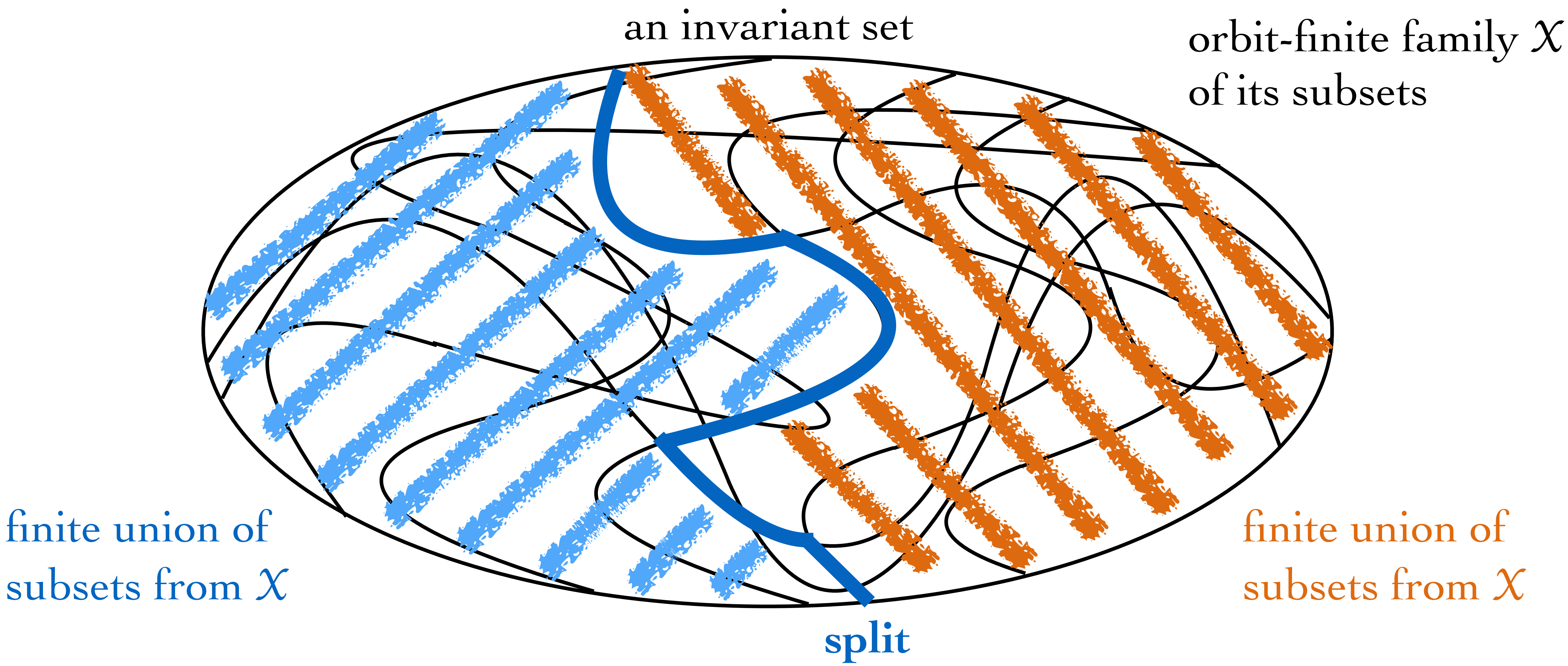
2) Deterministic collapse

Key tool: splitting lemma (+ orbit-finite Myhill-Nerode)



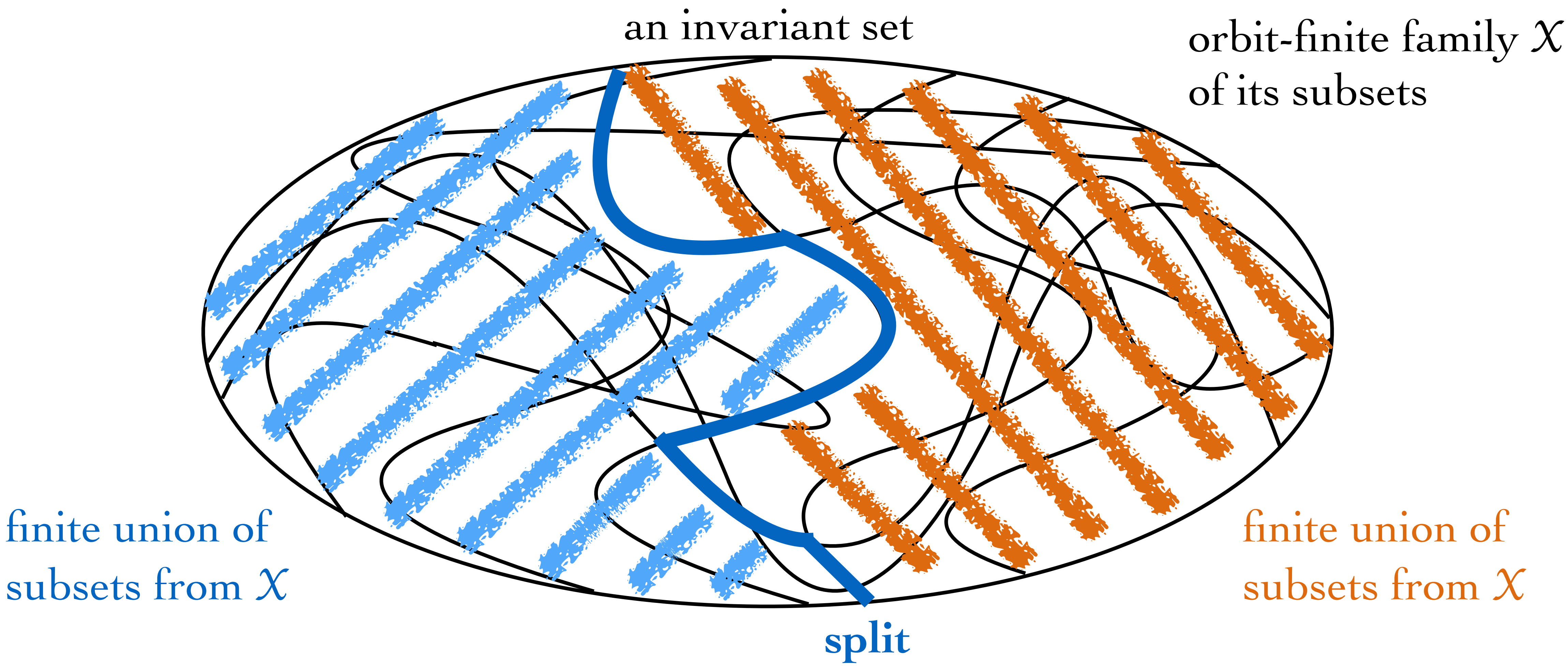
2) Deterministic collapse

Key tool: splitting lemma (+ orbit-finite Myhill-Nerode)



2) Deterministic collapse

Key tool: splitting lemma (+ orbit-finite Myhill-Nerode)



Splitting Lemma: each orbit-finite family $\mathcal{X}$ admits only orbit-finitely splits

3) Commutative images

Orbit-finite rational (regular) expressions:

$$\Big(\bigcup_{a,b\in\mathbb{A},\ a\neq b} ab\Big)^* \ \big(\varepsilon \cup \bigcup_{a\in\mathbb{A}} a\big)$$

3) Commutative images

Orbit-finite rational (regular) expressions:

$$\Big(\bigcup_{a,b\in\mathbb{A},\ a\neq b} ab\Big)^* \ (\varepsilon \cup \bigcup_{a\in\mathbb{A}} a) \qquad\qquad \bigcup_{a\in\mathbb{A}} a\Big(\bigcup_{b\in \mathrm{A}\setminus\{a\}} b\Big)^* a$$

3) Commutative images

Orbit-finite rational (regular) expressions:

$$\Big(\bigcup_{a,b\in\mathbb{A},\ a\neq b} ab\Big)^* \big(\varepsilon \cup \bigcup_{a\in\mathbb{A}} a\big) \qquad \bigcup_{a\in\mathbb{A}} a\Big(\bigcup_{b\in \mathrm{A}\setminus\{a\}} b\Big)^* a$$

Kleene fails: "each two neighbouring letters are different" is not rational.

3) Commutative images

Orbit-finite rational (regular) expressions:

$$\Big(\bigcup_{a,b\in\mathbb{A},\ a\neq b} ab\Big)^* \ (\varepsilon \cup \bigcup_{a\in\mathbb{A}} a) \qquad\qquad \bigcup_{a\in\mathbb{A}} a\Big(\bigcup_{b\in \mathrm{A}\setminus\{a\}} b\Big)^* a$$

Kleene fails: “each two neighbouring letters are different” is not rational.

Parikh helps!

3) Commutative images

Orbit-finite rational (regular) expressions:

$$\Big(\bigcup_{a,b \in \mathbb{A},\ a \neq b} ab \Big)^* \ (\varepsilon \cup \bigcup_{a \in \mathbb{A}} a) \qquad \bigcup_{a \in \mathbb{A}} a \Big(\bigcup_{b \in \mathrm{A} \setminus \{a\}} b \Big)^* a$$

Kleene fails: "each two neighbouring letters are different" is not rational.

Parikh helps!

Commutative image: $\mathbb{A}^* \to \mathbb{N}^{\mathbb{A}}$

data vectors

3) Commutative images

Orbit-finite rational (regular) expressions:

$$\Big(\bigcup_{a,b \in \mathbb{A},\ a \neq b} ab \Big)^* \ (\varepsilon \cup \bigcup_{a \in \mathbb{A}} a) \qquad\qquad \bigcup_{a \in \mathbb{A}} a \Big(\bigcup_{b \in \mathrm{A} \setminus \{a\}} b \Big)^* a$$

Kleene fails: "each two neighbouring letters are different" is not rational.

Parikh helps!

Commutative image: $\mathbb{A}^* \to \mathbb{N}^{\mathbb{A}}$
data vectors

Interpret rational expressions commutatively

3) Commutative images

Orbit-finite rational (regular) expressions:

$$\Big(\bigcup_{a,b \in \mathbb{A},\ a \neq b} ab \Big)^* \ (\varepsilon \cup \bigcup_{a \in \mathbb{A}} a) \qquad \bigcup_{a \in \mathbb{A}} a \Big(\bigcup_{b \in \mathrm{A} \setminus \{a\}} b \Big)^* a$$

Kleene fails: "each two neighbouring letters are different" is not rational.

Parikh helps!

Commutative image: $\mathbb{A}^* \to \mathbb{N}^{\mathbb{A}}$
data vectors

Interpret rational expressions commutatively

Commutative image of "each two neighbouring letters are different" is rational, being commutatively equivalent to

3) Commutative images

Orbit-finite rational (regular) expressions:

$$\Big(\bigcup_{a,b\in\mathbb{A},\ a\neq b} ab\Big)^* \ (\varepsilon \cup \bigcup_{a\in\mathbb{A}} a) \qquad\qquad \bigcup_{a\in\mathbb{A}} a\Big(\bigcup_{b\in A\setminus\{a\}} b\Big)^* a$$

Kleene fails: "each two neighbouring letters are different" is not rational.

Parikh helps!

Commutative image: $\mathbb{A}^* \to \mathbb{N}^{\mathbb{A}}$
data vectors

Interpret rational expressions commutatively

Commutative image of "each two neighbouring letters are different" is rational, being commutatively equivalent to

$$a_1 \neq a_2 \overset{?}{\neq} a_3 \neq a_4 \overset{?}{\neq} a_5 \neq a_6 \overset{?}{\neq} \ldots$$

3) Commutative images

Orbit-finite rational (regular) expressions:

$$\Big(\bigcup_{a,b\in\mathbb{A},\ a\neq b} ab\Big)^{*}\ \big(\varepsilon \cup \bigcup_{a\in\mathbb{A}} a\big) \qquad\qquad \bigcup_{a\in\mathbb{A}} a\Big(\bigcup_{b\in A\setminus\{a\}} b\Big)^{*} a$$

Kleene fails: "each two neighbouring letters are different" is not rational.

Parikh helps!

Commutative image: $\mathbb{A}^* \to \mathbb{N}^{\mathbb{A}}$
data vectors

Interpret rational expressions commutatively

Commutative image of "each two neighbouring letters are different" is rational, being commutatively equivalent to

$$a_1 \neq a_2 \neq a_3 \neq a_4 \neq a_5 \neq a_6 \neq \ldots$$

3) Commutative images

Orbit-finite rational (regular) expressions:

$$\Big(\bigcup_{a,b \in \mathbb{A},\ a \neq b} ab \Big)^* \ (\varepsilon \cup \bigcup_{a \in \mathbb{A}} a) \qquad\qquad \bigcup_{a \in \mathbb{A}} a \Big(\bigcup_{b \in A \setminus \{a\}} b \Big)^* a$$

Kleene fails: "each two neighbouring letters are different" is not rational.

Parikh helps!

Commutative image: $\mathbb{A}^* \to \mathbb{N}^{\mathbb{A}}$
data vectors

Interpret rational expressions commutatively

Commutative image of "each two neighbouring letters are different" is rational, being commutatively equivalent to

$$a_1 \neq a_2 \neq a_3 \neq a_4 \neq a_5 \neq a_6 \overset{?}{\neq} \ldots$$

3) Commutative images

Orbit-finite rational (regular) expressions:

$$\Big(\bigcup_{a,b\in\mathbb{A},\ a\neq b} ab\Big)^* \ (\varepsilon \cup \bigcup_{a\in\mathbb{A}} a) \qquad\qquad \bigcup_{a\in\mathbb{A}} a\Big(\bigcup_{b\in A\setminus\{a\}} b\Big)^* a$$

Kleene fails: “each two neighbouring letters are different” is not rational.

Parikh helps!

Commutative image: $\mathbb{A}^* \to \mathbb{N}^{\mathbb{A}}$
data vectors

Interpret rational expressions commutatively

Commutative image of “each two neighbouring letters are different” is rational, being commutatively equivalent to

$$a_1 \neq a_2 \neq a_3 \neq a_4 \neq a_5 \neq a_6 \overset{?}{\neq} \ldots$$

and even semi-linear

3) Commutative images

Theorem (Hofman, Juzepczuk, L., Pattathurajan 2021):
Commutative image of 1-NRA are rational, but not necessarily semi-linear.

3) Commutative images

Theorem (Hofman, Juzepczuk, L., Pattathurajan 2021):
Commutative image of 1-NRA are rational, but not necessarily semi-linear.

Likewise for 1-CFG.

3) Commutative images

Theorem (Hofman, Juzepczuk, L., Pattathurajan 2021):
Commutative image of 1-NRA are rational, but not necessarily semi-linear.

Likewise for 1-CFG.

Conjecture: Commutative images of all of NRA are rational.

3) Commutative images

Theorem (Hofman, Juzepczuk, L., Pattathurajan 2021**):**
Commutative image of 1-NRA are rational, but not necessarily semi-linear.

Likewise for 1-CFG.

Conjecture: Commutative images of all of NRA are rational.

Key tool: sufficient condition for Hamiltonian cycle
in a strongly connected directed graphs

4) Single-use registers

Bojańczyk, Stefański 2020

Idea: Register disappears (becomes $\perp$) after usage. Consider DRA.

4) Single-use registers

Bojańczyk, Stefański 2020

Idea: Register disappears (becomes $\bot$) after usage. Consider DRA.

Example: ''at most 3 different data values''

4) Single-use registers

Bojańczyk, Stefański 2020

Idea: Register disappears (becomes $\perp$) after usage. Consider DRA.

Example: "at most 3 different data values"

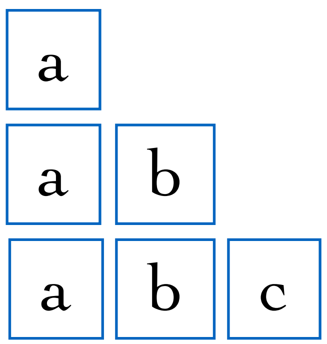
a		
a	b	
a	b	c

4) Single-use registers

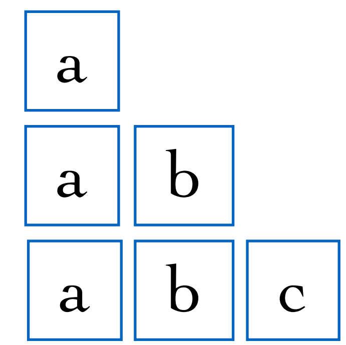
Bojańczyk, Stefański 2020

Idea: Register disappears (becomes $\bot$) after usage. Consider DRA.

Example: "at most 3 different data values"



- if input data equals a $\longrightarrow$

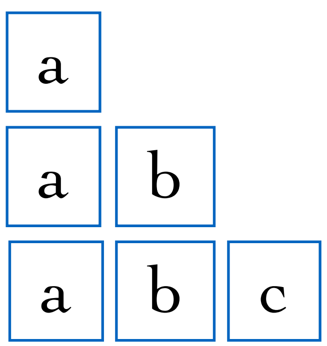


4) Single-use registers

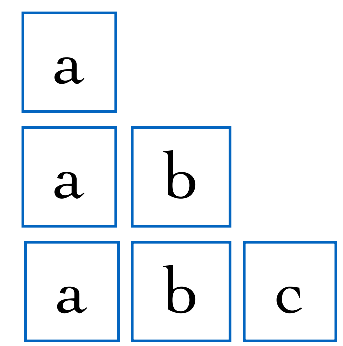
Bojańczyk, Stefański 2020

Idea: Register disappears (becomes $\bot$) after usage. Consider DRA.

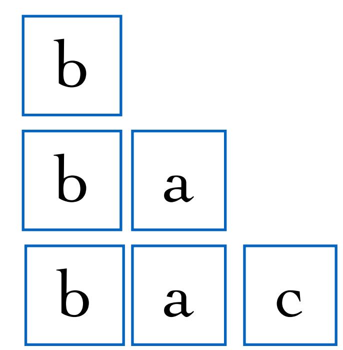
Example: "at most 3 different data values"



- if input data equals a $\longrightarrow$



- if input data equals b $\longrightarrow$

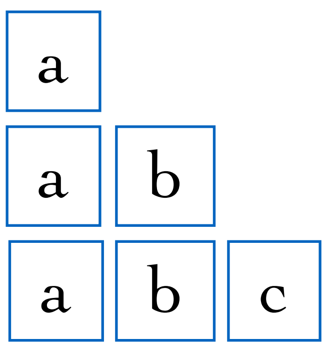


4) Single-use registers

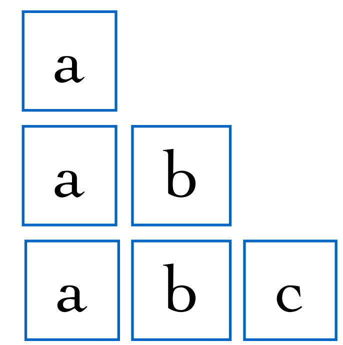
Bojańczyk, Stefański 2020

Idea: Register disappears (becomes $\bot$) after usage. Consider DRA.

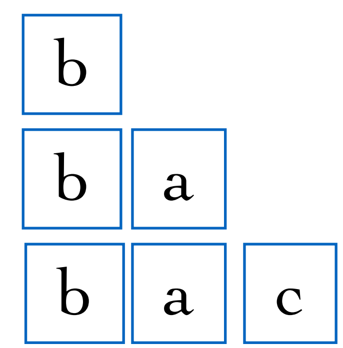
Example: "at most 3 different data values"



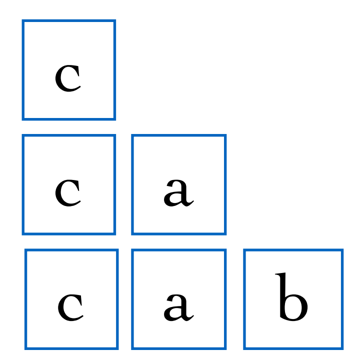
- if input data equals a $\longrightarrow$



- if input data equals b $\longrightarrow$



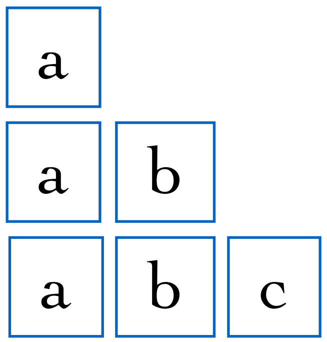
- if input data equals c $\longrightarrow$



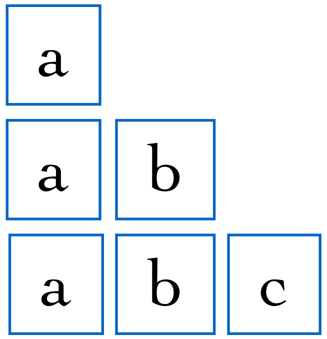
Bojańczyk, Stefański 2020

Idea: Register disappears (becomes ⊥) after usage. Consider DRA.

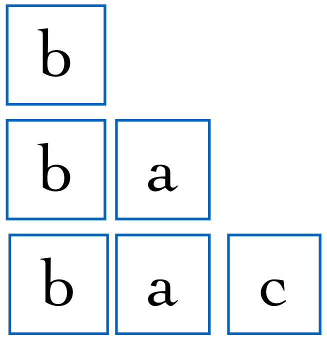
Example: "at most 3 different data values"



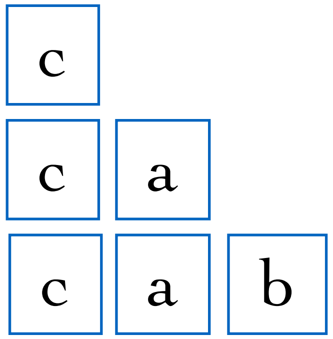
- if input data equals a ⟶



- if input data equals b ⟶



- if input data equals c ⟶



- otherwise reject

4) Single-use registers

Starting point:

orbit-finite monoids $\subset$ 1-DRA $\subset$ 2-DRA

4) Single-use registers

Starting point:

orbit-finite monoids $\subset$ 1-DRA $\subset$ 2-DRA

Theorem (Bojańczyk, Stefański 2020):

orbit-finite monoids = single-use 1-DRA = single-use 2-DRA

4) Single-use registers

Starting point:

orbit-finite monoids $\subset$ 1-DRA $\subset$ 2-DRA

Theorem (Bojańczyk, Stefański 2020**):**

orbit-finite monoids = single-use 1-DRA = single-use 2-DRA

Key tool: generalise the standard transformation of 2-way to 1-way automata, to register automata

I. Introduction to register automata

II. Some recent advances

1) Deterministic separability
2) Deterministic collapse
3) Commutative images
4) Single-use registers

thank you!

Oligomorphic atoms

a structure $\mathbb{A}$ is **oligomorphic**

if

$\mathbb{A}^{(n)}$ split into finitely many orbits for every n.

Oligomorphic atoms

a structure $\mathbb{A}$ is **oligomorphic**

if

$\mathbb{A}^{(n)}$ split into finitely many orbits for every n.

Example: for atoms $(\mathbb{Q}, \leq)$, $\text{atoms}^{(n)}$ has $n!$ orbits

Oligomorphic atoms

a structure $\mathbb{A}$ is **oligomorphic**

if

$\mathbb{A}^{(n)}$ split into finitely many orbits for every n.

Example: for atoms $(\mathbb{Q}, \leq)$, atoms$^{(n)}$ has n! orbits

Example: for atoms $(\mathbb{Q}, \leq, +1)$, atoms$^{(2)}$ has infinitely many orbits

$(7,6\frac{1}{3})$ $(7,7\frac{1}{3})$ $(7,8)$ $(7,8\frac{1}{3})$ ...

Oligomorphic atoms

a structure $\mathbb{A}$ is **oligomorphic**

if

$\mathbb{A}^{(n)}$ split into finitely many orbits for every n.

Example: for atoms $(\mathbb{Q}, \leq)$, $\text{atoms}^{(n)}$ has n! orbits

Example: for atoms $(\mathbb{Q}, \leq, +1)$, $\text{atoms}^{(2)}$ has infinitely many orbits

$(7, 6\tfrac{1}{3}) \quad (7, 7\tfrac{1}{3}) \quad (7, 8) \quad (7, 8\tfrac{1}{3}) \quad \ldots$

Theorem: orbit-finite sets are stable under Cartesian products and subsets

Oligomorphic atoms

a structure $\mathbb{A}$ is **oligomorphic**

if

$\mathbb{A}^{(n)}$ split into finitely many orbits for every n.

Example: for atoms $(\mathbb{Q}, \leq)$, atoms$^{(n)}$ has n! orbits

Example: for atoms $(\mathbb{Q}, \leq, +1)$, atoms$^{(2)}$ has infinitely many orbits

(7,6⅓) (7,7⅓) (7,8) (7,8⅓) ...

Theorem: orbit-finite sets are stable under Cartesian products and subsets

Theorem:

invariant subsets of $\mathbb{A}^n$ = **FO** definable subsets of $\mathbb{A}^n$

Homogeneous atoms (relational case)

Homogeneous atoms (relational case)

a relational structure $\mathbb{A}$ is homogeneous

if

Homogeneous atoms (relational case)

a relational structure $\mathbb{A}$ is homogeneous

if

every isomorphism of **finite induced substructures** of $\mathbb{A}$

extends to an automorphism of the whole structure

Homogeneous atoms (relational case)

a relational structure $\mathbb{A}$ is homogeneous

if

every isomorphism of **finite induced substructures** of $\mathbb{A}$

extends to an automorphism of the whole structure

Example: $(\mathbb{Q}, \leq)$

Homogeneous atoms (relational case)

a relational structure $\mathbb{A}$ is homogeneous

if

every isomorphism of **finite induced substructures** of $\mathbb{A}$

extends to an automorphism of the whole structure

Example: $(\mathbb{Q}, \leq)$

Homogeneous atoms (relational case)

a relational structure $\mathbb{A}$ is homogeneous

if

every isomorphism of **finite induced substructures** of $\mathbb{A}$

extends to an automorphism of the whole structure

Example: $(\mathbb{Q}, \leq)$

Homogeneous atoms (relational case)

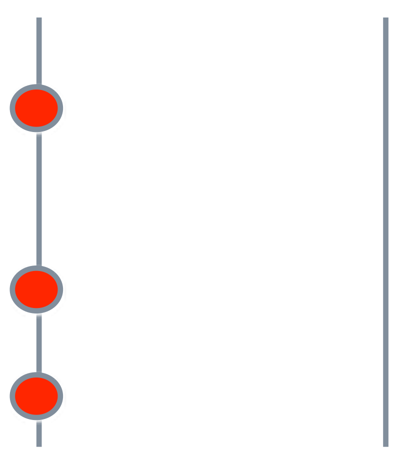
a relational structure $\mathbb{A}$ is homogeneous

if

every isomorphism of **finite induced substructures** of $\mathbb{A}$

extends to an automorphism of the whole structure

Example: $(\mathbb{Q}, \leq)$



Homogeneous atoms (relational case)

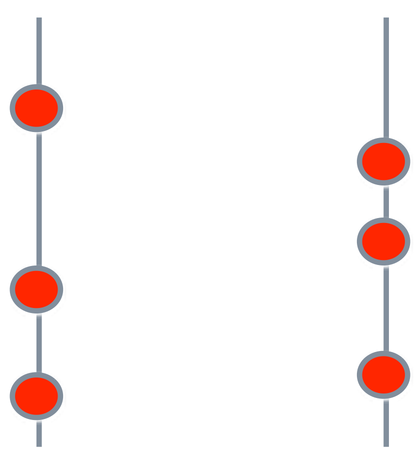
a relational structure $\mathbb{A}$ is homogeneous

if

every isomorphism of **finite induced substructures** of $\mathbb{A}$

extends to an automorphism of the whole structure

Example: $(\mathbb{Q}, \leq)$



Homogeneous atoms (relational case)

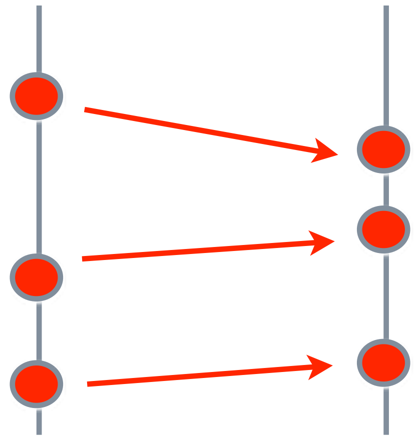
a relational structure $\mathbb{A}$ is homogeneous

if

every isomorphism of **finite induced substructures** of $\mathbb{A}$

extends to an automorphism of the whole structure

Example: $(\mathbb{Q}, \leq)$



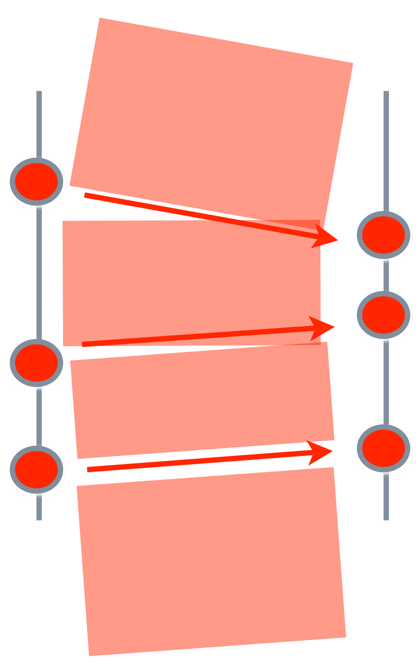
Homogeneous atoms (relational case)

a relational structure $\mathbb{A}$ is homogeneous

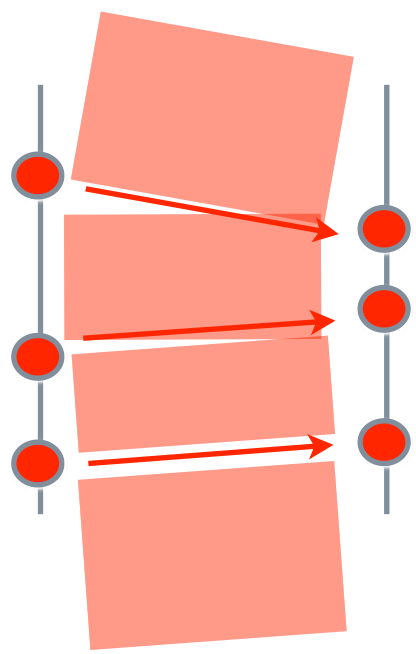
if

every isomorphism of **finite induced substructures** of $\mathbb{A}$
extends to an automorphism of the whole structure

Example: $(\mathbb{Q}, \leq)$

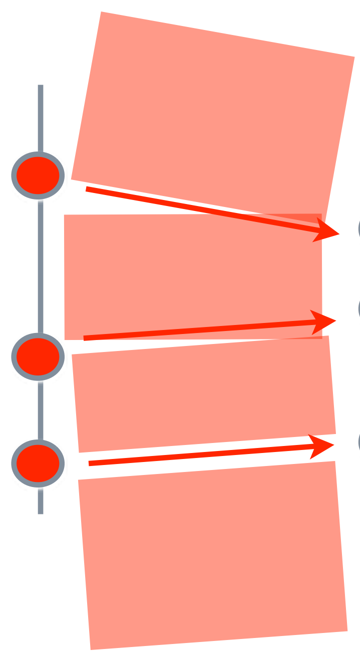


Homogeneous atoms (relational case)



Examples:

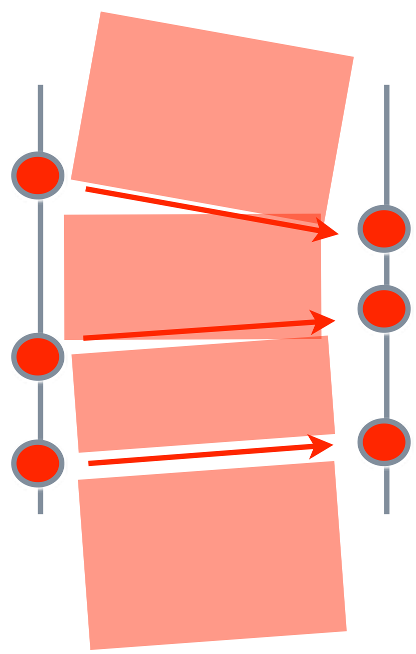
Homogeneous atoms (relational case)



Examples:

total order atoms $(\mathbb{Q}, <)$

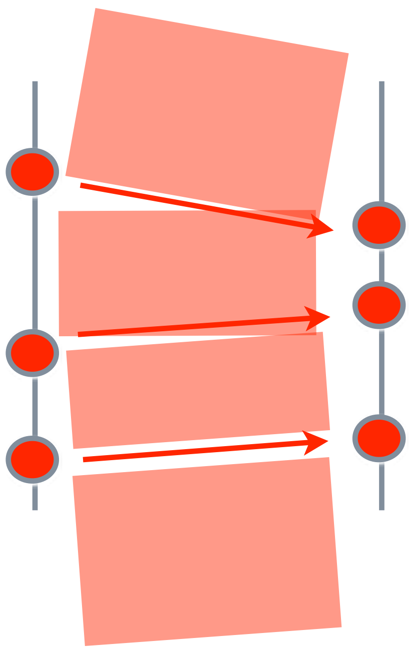
Homogeneous atoms (relational case)



Examples:

total order atoms $(\mathbb{Q}, <)$
integer atoms $(\mathbb{Z}, <)$

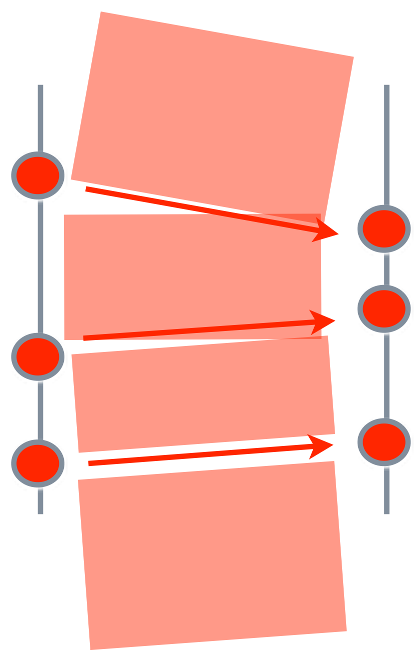
Homogeneous atoms (relational case)



Examples:

total order atoms $(\mathbb{Q}, <)$
~~integer atoms $(\mathbb{Z}, <)$~~

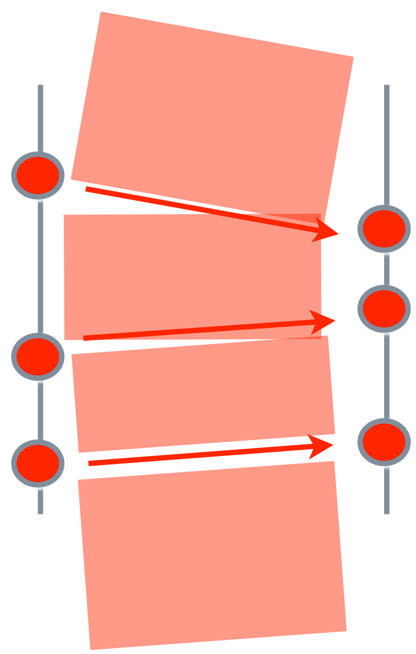
Homogeneous atoms (relational case)



Examples:

total order atoms $(\mathbb{Q}, <)$
~~integer atoms $(\mathbb{Z}, <)$~~
$(\mathbb{Q}, <, +1)$

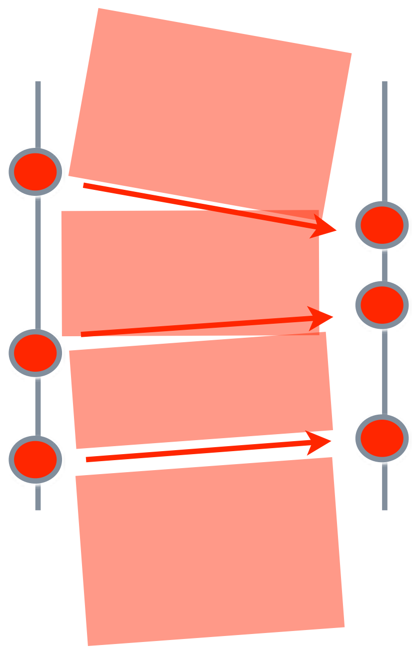
Homogeneous atoms (relational case)



Examples:

total order atoms $(\mathbb{Q}, <)$
~~integer atoms $(\mathbb{Z}, <)$~~
~~$(\mathbb{Q}, <, +1)$~~

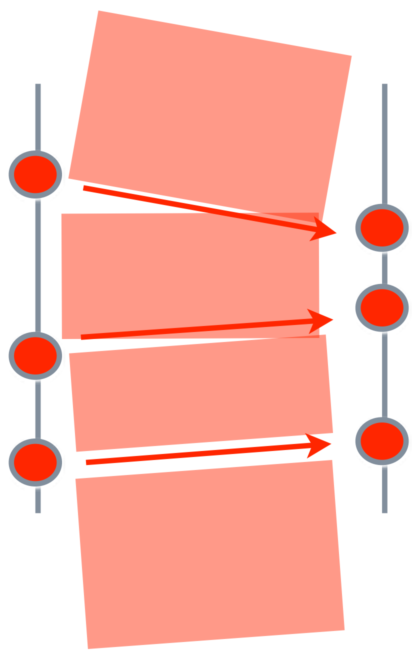
Homogeneous atoms (relational case)



Examples:

total order atoms $(\mathbb{Q}, <)$
~~integer atoms $(\mathbb{Z}, <)$~~
~~$(\mathbb{Q}, <, +1)$~~
equality atoms $(\mathbb{N}, =)$

Homogeneous atoms (relational case)

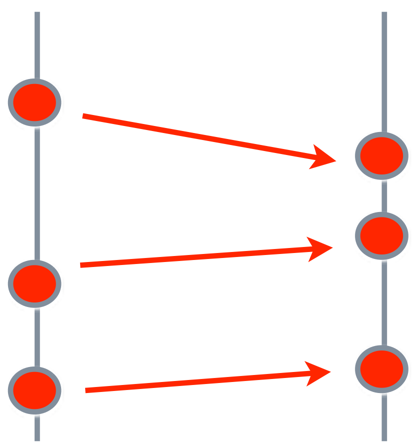


Examples:

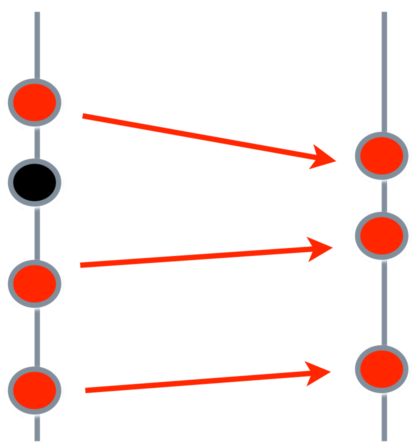
total order atoms ($\mathbb{Q}$, <)
~~integer atoms ($\mathbb{Z}$, <)~~
~~($\mathbb{Q}$, <, +1)~~
equality atoms ($\mathbb{N}$, =)
random graph
...

random graph = countable infinite graph yielded almost surely if every pair of nodes is connected by an edge with independent probability ⅓

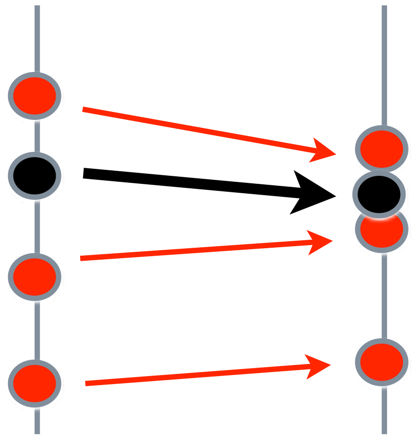
Homogeneous atoms (relational case)



Homogeneous atoms (relational case)

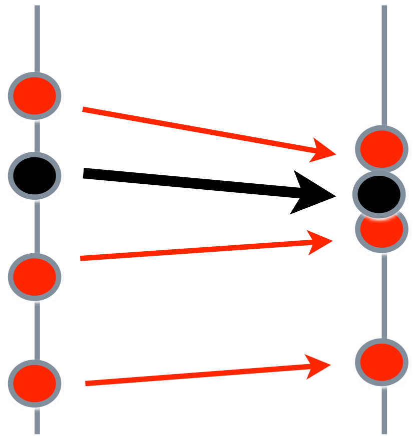


Homogeneous atoms (relational case)

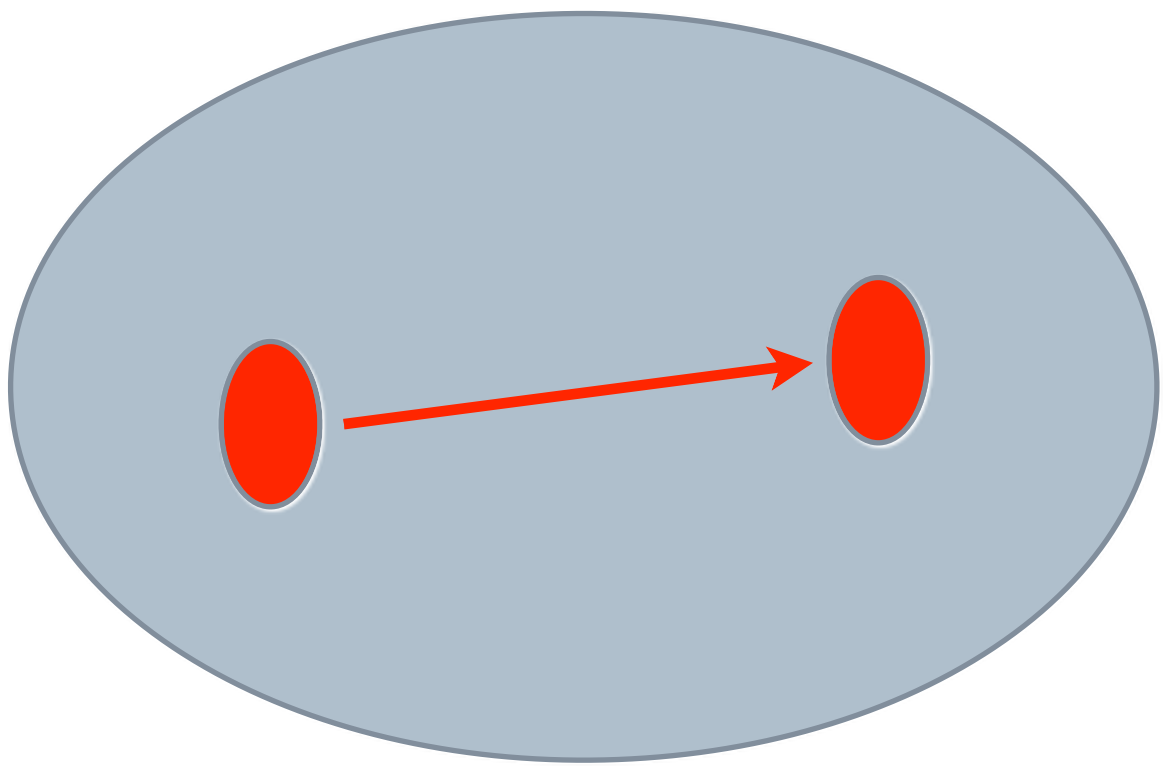


extension property

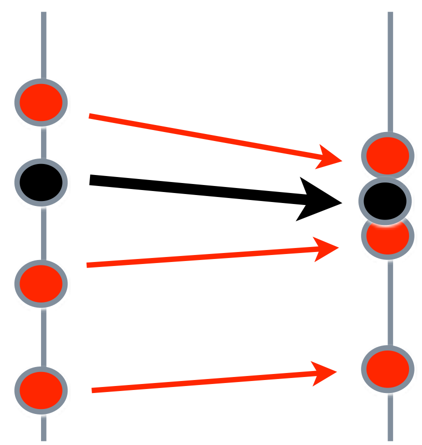
Homogeneous atoms (relational case)



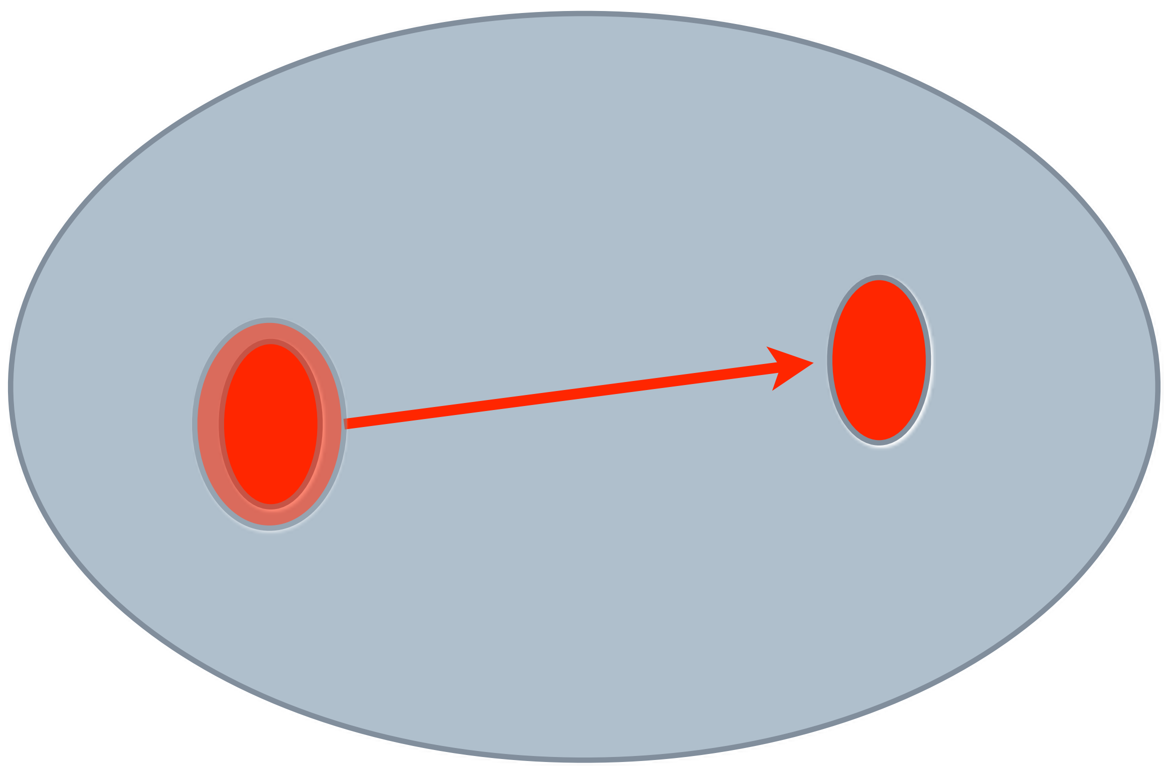
extension property



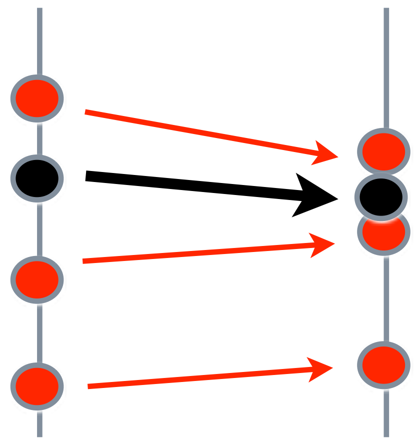
Homogeneous atoms (relational case)



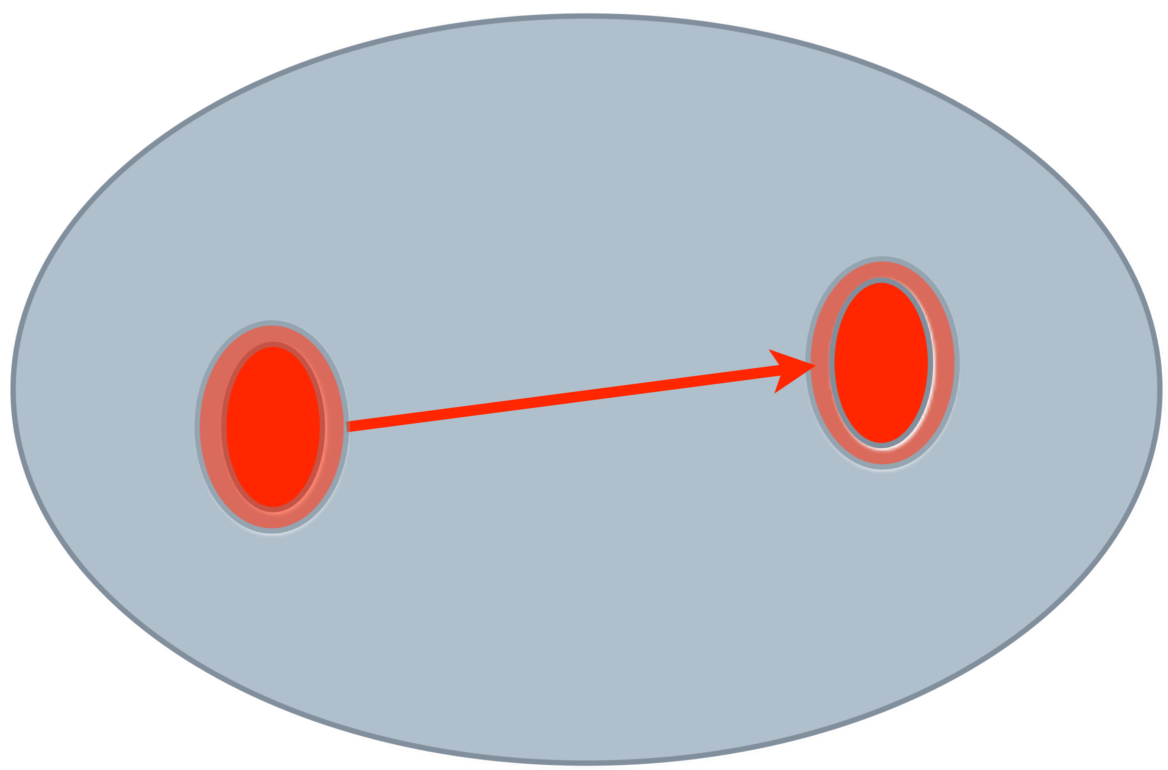
extension property



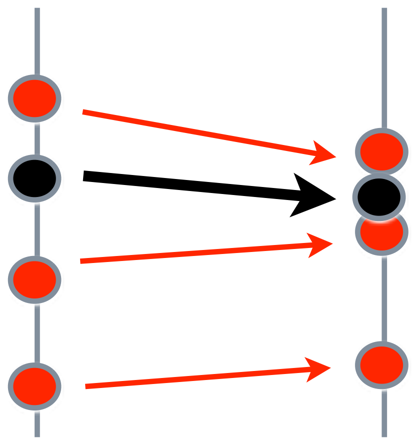
Homogeneous atoms (relational case)



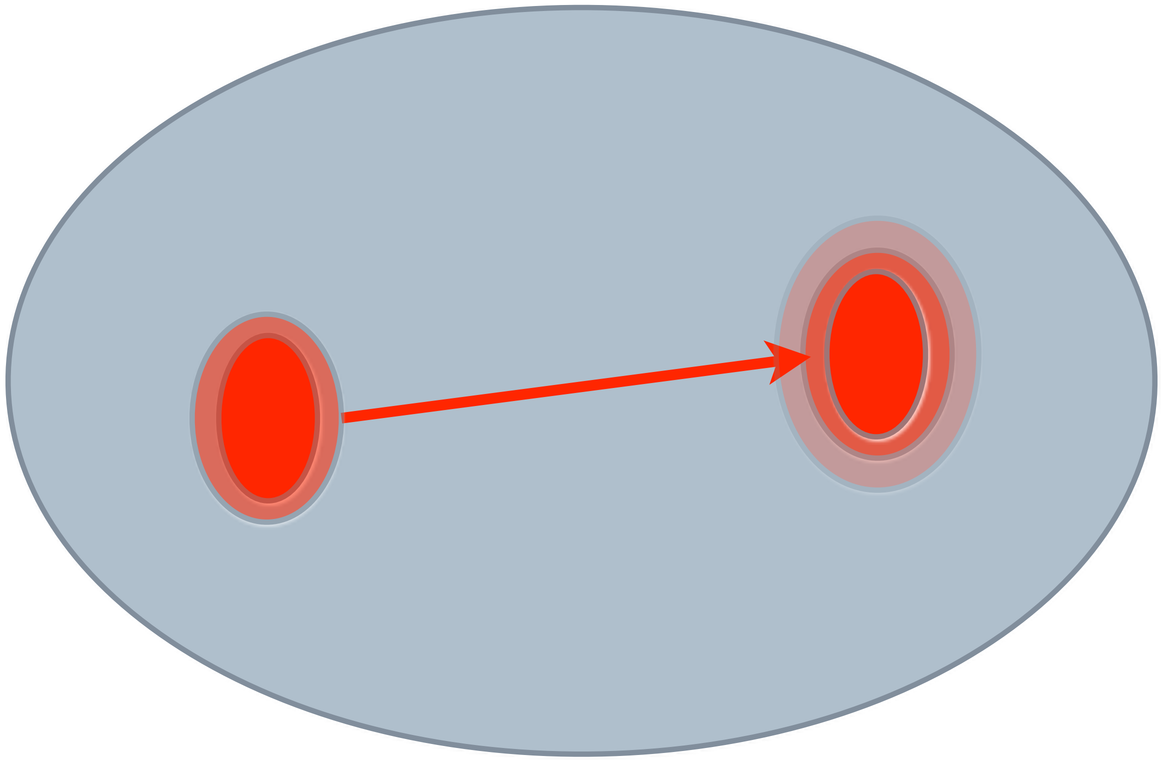
extension property



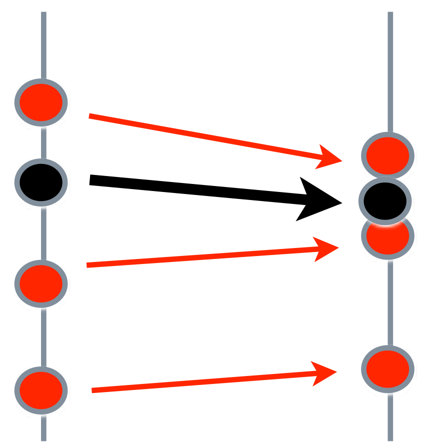
Homogeneous atoms (relational case)



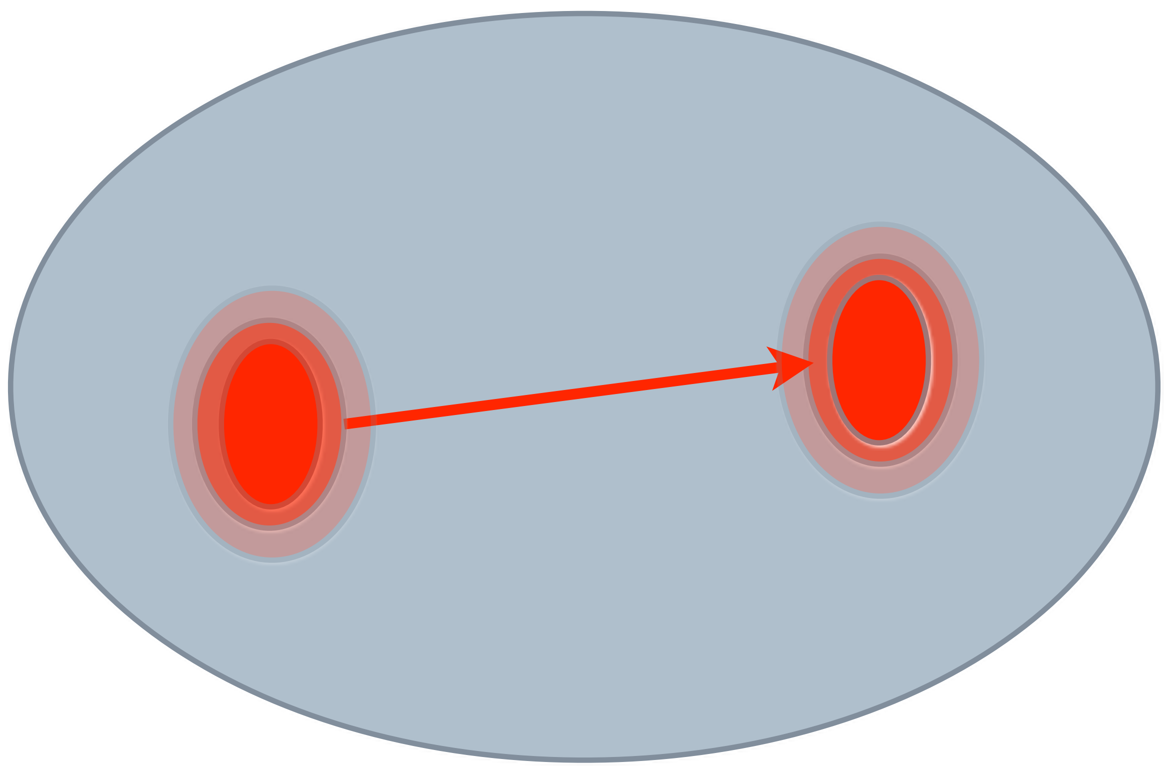
extension property



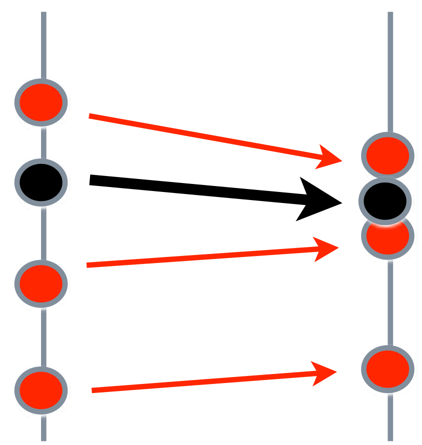
Homogeneous atoms (relational case)



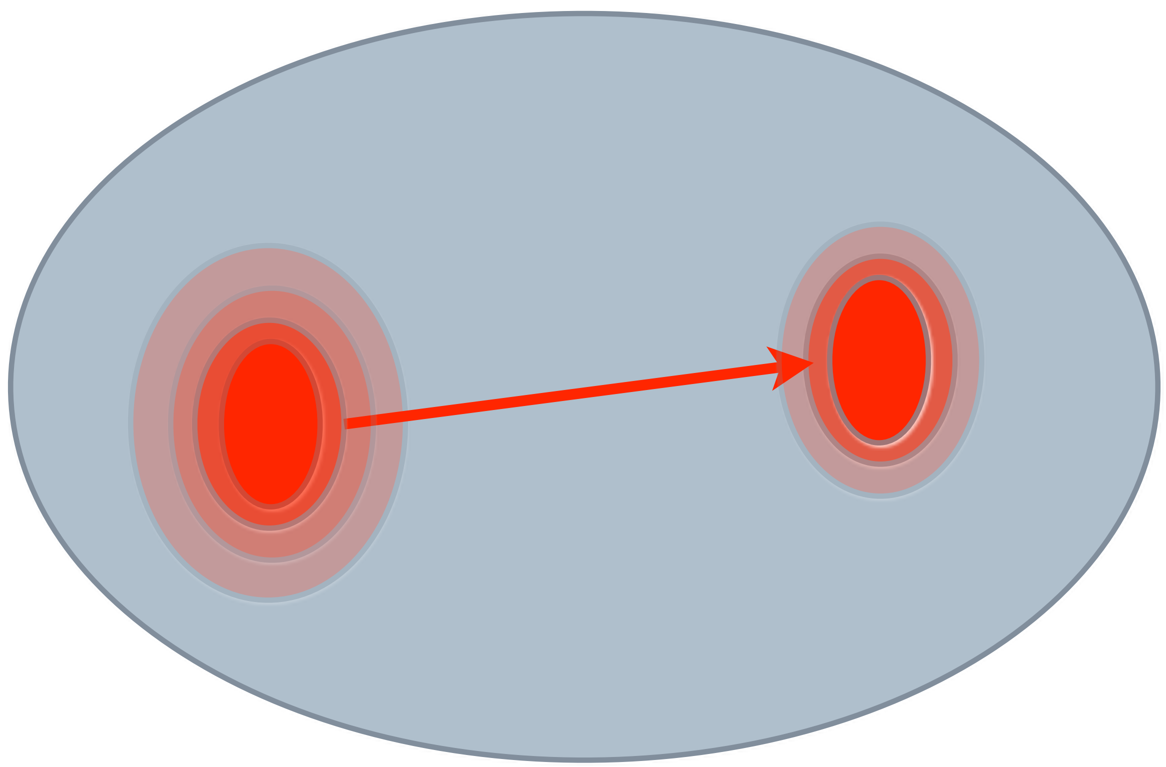
extension property



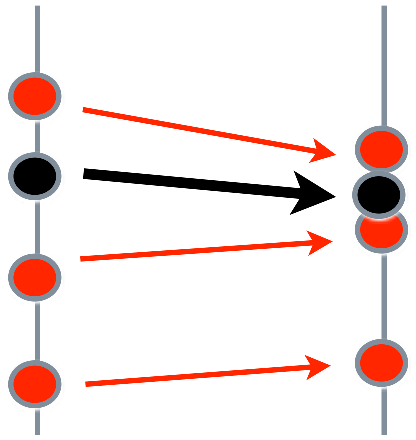
Homogeneous atoms (relational case)



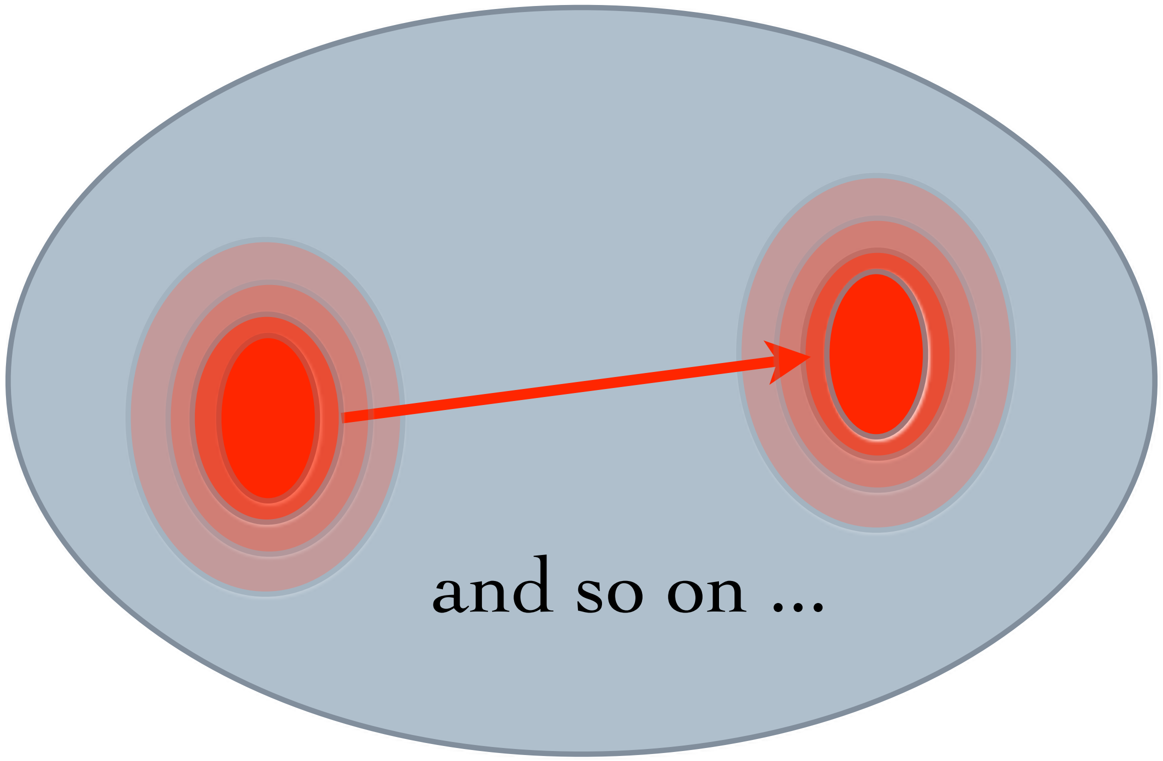
extension property



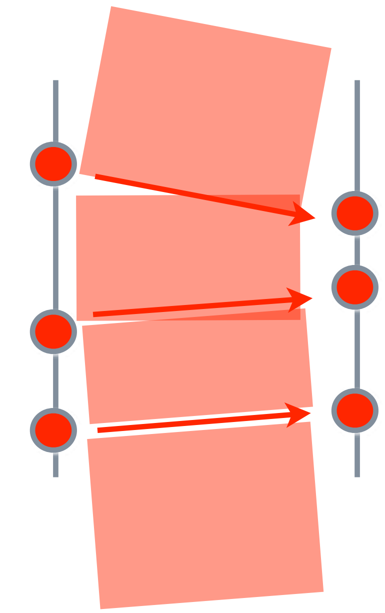
Homogeneous atoms (relational case)



extension property



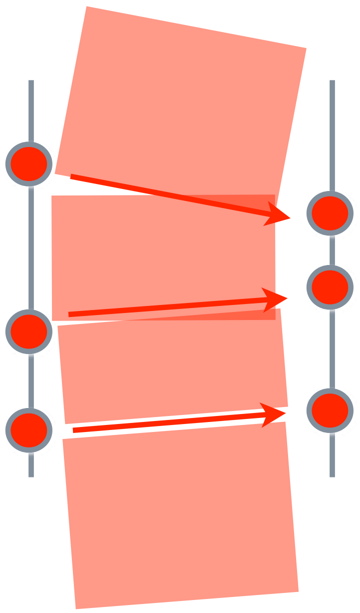
Quantifier elimination



Observation: When atoms are homogeneous

two tuples in atoms$^{(n)}$ are in the same orbit	iff	the induce generate isomorphic substructures

Quantifier elimination



Observation: When atoms are homogeneous

two tuples in atoms$^{(n)}$ are in the same orbit **iff** the induce generate isomorphic substructures

Corollary:

invariant subsets of $\mathbb{A}^n$ = **quantifier-free** definable subsets of $\mathbb{A}^n$