Turing machines over infinite alphabets

Sławomir Lasota
University of Warsaw

joint work with
Mikołaj Bojańczyk, Bartek Klin, Joanna Ochremiak, Szymon Toruńczyk

Dagstuhl seminar 14141, 2014.04.03

Turing machines over infinite alphabets
...but finite up to permutation (orbit-finite)

Sławomir Lasota
University of Warsaw

joint work with
Mikołaj Bojańczyk, Bartek Klin, Joanna Ochremiak, Szymon Toruńczyk

Dagstuhl seminar 14141, 2014.04.03

computation in sets with atoms

computation in sets with atoms

sets with urelements

permutation models
[Mostowski '39]

sets with symmetry

nominal sets
[Gabbay, Pitts '99]

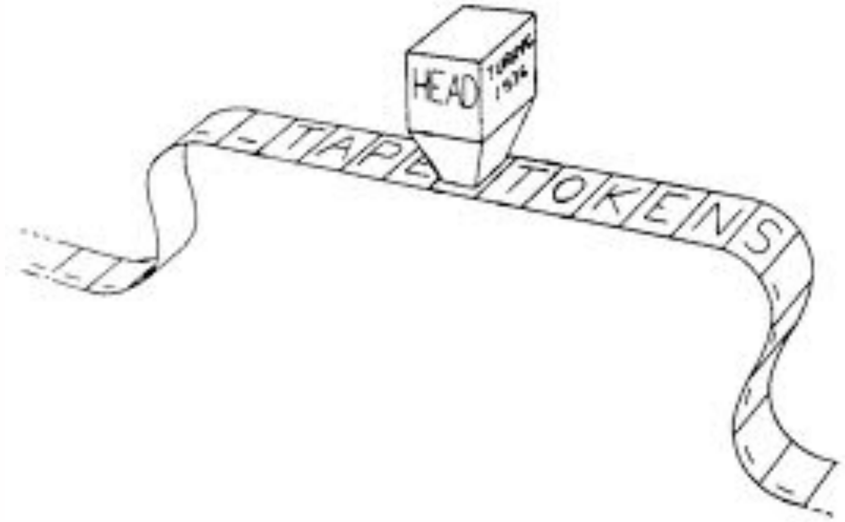
Fraenkel-Mostowski sets

named sets
[Pistore, Montanari '97]

nominal G-sets

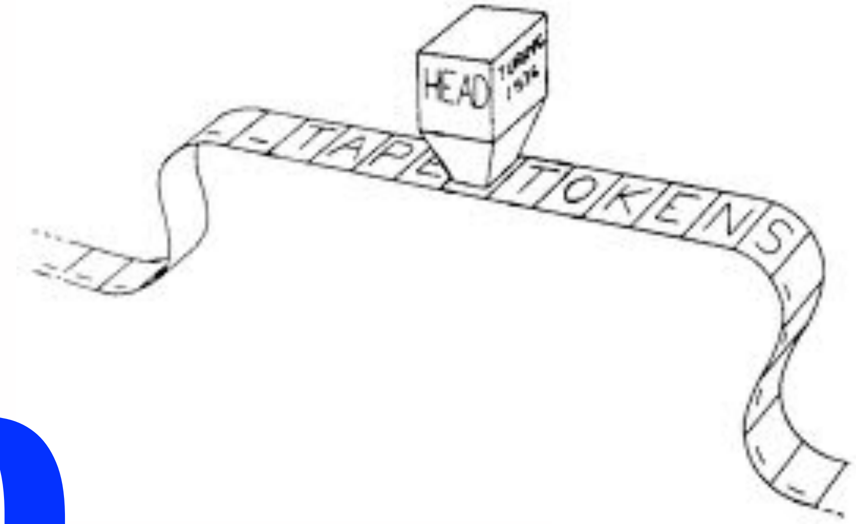
hereditarily finitely-supported sets

Turing machines
(any atoms)



- tape alphabet A
- states Q
- subset $\delta \subseteq Q \times A \times Q \times A \times \{\leftarrow, \rightarrow, \downarrow\}$
- subsets I, $F \subseteq Q$

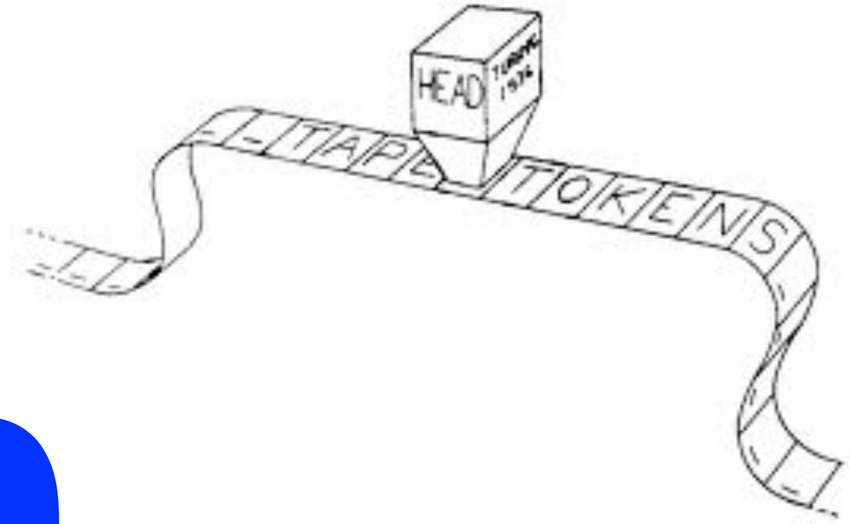
Turing machines
(any atoms)



- tape alphabet A
- states Q
- subset $\delta \subseteq Q \times A \times Q \times A \times \{\leftarrow, \rightarrow, \downarrow\}$
- subsets $I, F \subseteq Q$

} orbit-finite sets instead of finite ones

Turing machines
(any atoms)

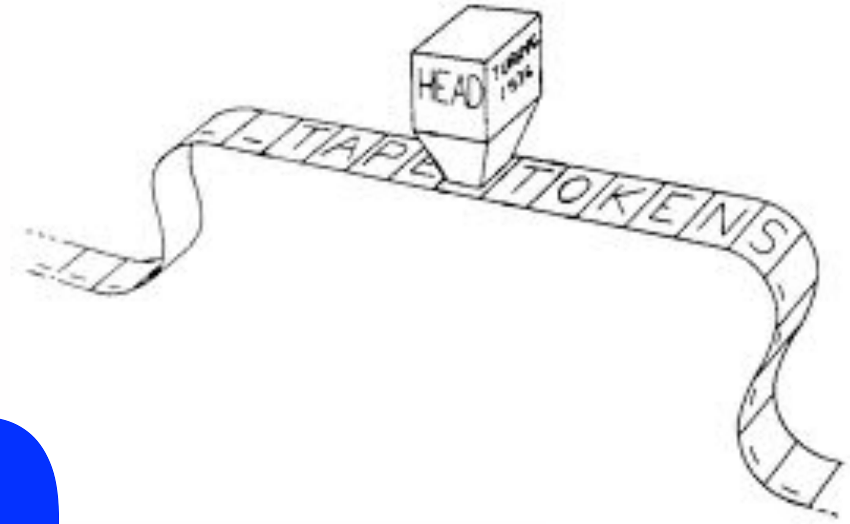


- tape alphabet A
- states Q
- subset $\delta \subseteq Q \times A \times Q \times A \times \{\leftarrow, \rightarrow, \downarrow\}$
- subsets I, F $\subseteq$ Q

} orbit-finite sets instead of finite ones

Configurations = $A^* \times Q \times A^*$

Turing machines
(any atoms)



- tape alphabet A
- states Q
- subset $\delta \subseteq Q \times A \times Q \times A \times \{\leftarrow, \rightarrow, \downarrow\}$
- subsets I, F $\subseteq Q$

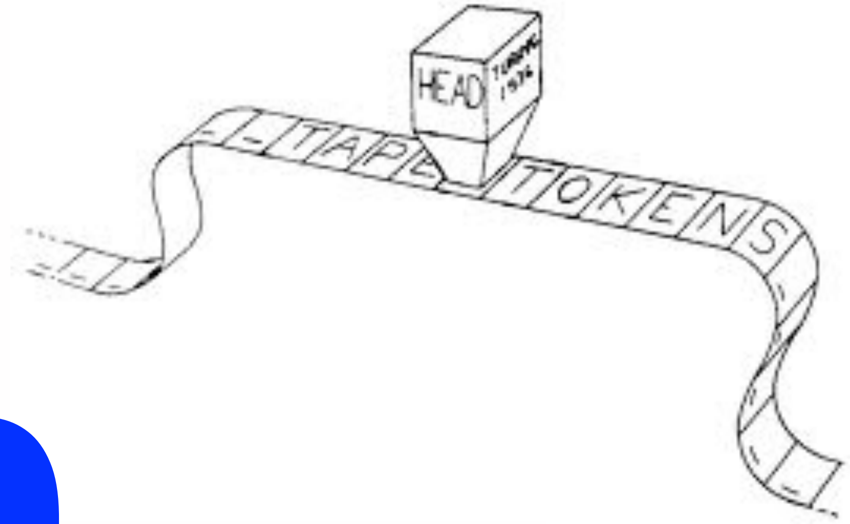
} orbit-finite sets instead of finite ones

Configurations = $A^* \times Q \times A^*$

Deterministic machines:

- $\delta : Q \times A \rightarrow Q \times A \times \{\leftarrow, \rightarrow, \downarrow\}$

Turing machines
(any atoms)



- tape alphabet A
- states Q
- subset $\delta \subseteq Q \times A \times Q \times A \times \{\leftarrow, \rightarrow, \downarrow\}$
- subsets I, F $\subseteq$ Q

} orbit-finite sets instead of finite ones

Configurations = $A^* \times Q \times A^*$

Deterministic machines:

- $\delta : Q \times A \rightarrow Q \times A \times \{\leftarrow, \rightarrow, \downarrow\}$

Consider only atoms (N, =) in this presentation

input alphabet: atoms

language:

tape alphabet:

states:

transitions:

input alphabet: atoms

atoms $\times$ (a finite set)

language:

tape alphabet:

states:

transitions:

input alphabet: atoms

language: "no atom appears twice":

$\{a_1 a_2 \ldots a_n \ : \ a_i \neq a_j \text{ when } i \neq j\}$

tape alphabet:

states:

transitions:

input alphabet: atoms

language: "no atom appears twice":

$\{a_1 a_2 \ldots a_n \ : \ a_i \neq a_j \text{ when } i \neq j\}$

tape alphabet: A = atoms ∪ {⊥}

states:

transitions:

input alphabet: atoms

language: "no atom appears twice":

$\{a_1 a_2 \ldots a_n \;:\; a_i \neq a_j \text{ when } i \neq j\}$

tape alphabet: A = atoms ∪ {⊥}

states: Q = atoms ∪ {start, accept, ret}

transitions:

input alphabet: atoms

language: "no atom appears twice":

$\{a_1 a_2 \ldots a_n \;:\; a_i \neq a_j \text{ when } i \neq j\}$

tape alphabet: A = atoms ∪ {⊥}

states: Q = atoms ∪ {start, accept, ret}

transitions: $\delta : Q \times A \to Q \times A \times \{\leftarrow, \rightarrow, \downarrow\}$

input alphabet: atoms

language: "no atom appears twice":

$\{a_1 a_2 \ldots a_n \ : \ a_i \neq a_j \text{ when } i \neq j\}$

tape alphabet: A = atoms ∪ {⊥}

states: Q = atoms ∪ {start, accept, ret}

transitions: δ : Q × A → Q × A × {←,→,↓}

δ(start, a) = (a, ⊥, →) a ∈ atoms

if in state start atom a is read from tape, goto state a, write ⊥ on tape, and move right

input alphabet: atoms

language: "no atom appears twice":

$\{a_1 a_2 \ldots a_n \ : \ a_i \neq a_j \text{ when } i \neq j\}$

tape alphabet: A = atoms ∪ {⊥}

states: Q = <u>atoms</u> ∪ {start, accept, ret}

transitions: δ : Q × A → Q × A × {←,→,↓}

δ(start, a) = (<u>a</u>, ⊥, →) a ∈ atoms

δ(<u>a</u>, b) = (<u>a</u>, b, →) a ≠ b, a, b ∈ atoms

if in state <u>a</u> atom b ≠ a is read from tape, stay in state <u>a</u>, write b on tape, and move right

input alphabet: atoms

language: "no atom appears twice":

$\{a_1 a_2 \ldots a_n \ : \ a_i \neq a_j \text{ when } i \neq j\}$

tape alphabet: A = atoms ∪ {⊥}

states: Q = atoms ∪ {start, accept, ret}

transitions: δ : Q × A → Q × A × {←,→,↓}

δ(start, a) =	(a, ⊥, →)	a ∈ atoms
δ(a, b) =	(a, b, →)	a ≠ b, a, b ∈ atoms
δ(a, B) =	(ret, B, ←)	a ∈ atoms

input alphabet: atoms

language: "no atom appears twice":

$\{a_1 a_2 \ldots a_n \ : \ a_i \neq a_j$ when $i \neq j\}$

tape alphabet: A = atoms ∪ {⊥}

states: Q = <u>atoms</u> ∪ {start, accept, ret}

transitions: $\delta : Q \times A \to Q \times A \times \{\leftarrow, \rightarrow, \downarrow\}$

δ(start, a) =	(<u>a</u>, ⊥, →)	a ∈ atoms
δ(<u>a</u>, b) =	(<u>a</u>, b, →)	a ≠ b, a, b ∈ atoms
δ(<u>a</u>, B) =	(ret, B, ←)	a ∈ atoms
δ(ret, a) =	(ret, a, ←)	a ∈ atoms

input alphabet: atoms

language: "no atom appears twice":

$\{a_1 a_2 \ldots a_n \;:\; a_i \neq a_j \text{ when } i \neq j\}$

tape alphabet: A = atoms ∪ {⊥}

states: Q = atoms ∪ {start, accept, ret}

transitions: δ : Q × A → Q × A × {←,→,↓}

δ(start, a) = (a, ⊥, →) a ∈ atoms

δ(a, b) = (a, b, →) a ≠ b, a, b ∈ atoms

δ(a, B) = (ret, B, ←) a ∈ atoms

δ(ret, a) = (ret, a, ←) a ∈ atoms

δ(ret, ⊥) = (start, ⊥, →)

input alphabet: atoms

language: "no atom appears twice": $\{a_1 a_2 \ldots a_n \; : \; a_i \neq a_j \text{ when } i \neq j\}$

tape alphabet: $A = \text{atoms} \cup \{\bot\}$

states: $Q = \underline{\text{atoms}} \cup \{\text{start}, \text{accept}, \text{ret}\}$

transitions: $\delta : Q \times A \to Q \times A \times \{\leftarrow, \rightarrow, \downarrow\}$

$\delta(\text{start}, a) = (\underline{a}, \bot, \rightarrow)$ $a \in \text{atoms}$

$\delta(\underline{a}, b) = (\underline{a}, b, \rightarrow)$ $a \neq b, \; a, b \in \text{atoms}$

$\delta(\underline{a}, B) = (\text{ret}, B, \leftarrow)$ $a \in \text{atoms}$

$\delta(\text{ret}, a) = (\text{ret}, a, \leftarrow)$ $a \in \text{atoms}$

$\delta(\text{ret}, \bot) = (\text{start}, \bot, \rightarrow)$

$\delta(\text{start}, B) = (\text{accept}, B, \rightarrow)$

Do TMs determinize?

Do TMs determinize?

Example alphabets:

- atoms
- ordered pairs of atoms
- unordered pairs of atoms
- unordered pairs of ordered pairs of atoms
- ordered triples of pairs of atoms modulo even number of flips

Do TMs determinize?

Example alphabets:

- atoms
- ordered pairs of atoms
- unordered pairs of atoms
- unordered pairs of ordered pairs of atoms
- ordered triples of pairs of atoms modulo even number of flips

} standard (first four)

Do TMs determinize?

Example alphabets:

- atoms
- ordered pairs of atoms
- unordered pairs of atoms
- unordered pairs of ordered pairs of atoms

} standard

- ordered triples of pairs of atoms modulo even number of flips

non-standard

alphabet: atoms

alphabet: atoms

a b a e d d c d f d g y h e u s e d f e r g f f e d s

alphabet: atoms

guess an atom
different than h

a b a e d d c d f d g y h e u s e d f e r g f f e d s

alphabet: atoms

guess an atom different than h

a b a e d d c d f d g y h e u s e d f e r g f f e d s

de-atomization:

alphabet: atoms

guess an atom
different than h

a b a e d d c d f d g y h e u s e d f e r g f f e d s

de-atomization:

- replacing atoms with binary encodings

a	1
b	101
e	1001
d	10001
c	...

alphabet: atoms

guess an atom different than h

a b a e d d c d f d g y h e u s e d f e r g f f e d s

de-atomization:

- replacing atoms with binary encodings
- atom-less simulation of computation of a machine

a	1
b	101
e	1001
d	10001
c	...

alphabet: atoms

guess an atom different than h

a b a e d d c d f d g y h e u s e d f e r g f f e d s

de-atomization:

- replacing atoms with binary encodings
- atom-less simulation of computation of a machine

a	1
b	101
e	1001
d	10001
c	...

Fact: TMs over this alphabet do determinize

alphabet: ordered pairs of atoms

$$(a, b) \in \text{atoms}^2 \qquad a \neq b$$

alphabet: ordered pairs of atoms

$$(a, b) \in \text{atoms}^2 \qquad a \neq b$$

- input word represents a directed graph

alphabet: ordered pairs of atoms

$$(a, b) \in \text{atoms}^2 \qquad a \neq b$$

- input word represents a directed graph
- nodes (atoms) can be computed using projections

$$(a, b) \mapsto a \qquad\qquad (a, b) \mapsto b$$

  and stored on the tape

- then any decidable property of directed graphs can be decided deterministically

alphabet: ordered pairs of atoms

$$(a, b) \in \text{atoms}^2 \qquad a \neq b$$

- input word represents a directed graph
- nodes (atoms) can be computed using projections

$$(a, b) \mapsto a \qquad\qquad (a, b) \mapsto b$$

  and stored on the tape

- then any decidable property of directed graphs can be decided deterministically

Fact: TMs over this alphabet do determinize

alphabet: unordered pairs of atoms

$$\{a, b\} \in \mathcal{P}_2(\text{atoms})$$

alphabet: unordered pairs of atoms

$$\{a, b\} \in \mathcal{P}_2(\text{atoms})$$

- input word represents an undirected graph

alphabet: unordered pairs of atoms

$\{a, b\} \in \mathcal{P}_2(\text{atoms})$

- input word represents an undirected graph
- can nodes (atoms) be computed?

- then any decidable property of undirected graphs can be decided deterministically

alphabet: unordered pairs of atoms

$$\{a, b\} \in \mathcal{P}_2(\text{atoms})$$

- input word represents an undirected graph
- can nodes (atoms) be computed?

$$\{a, b\} \mapsto a$$

- then any decidable property of undirected graphs can be decided deterministically

alphabet: unordered pairs of atoms

$$\{a, b\} \in \mathcal{P}_2(\text{atoms})$$

- input word represents an undirected graph
- can nodes (atoms) be computed?

$$\{a, b\} \mapsto a$$

- then any decidable property of undirected graphs can be decided deterministically

alphabet: unordered pairs of atoms

$$\{a, b\} \in \mathcal{P}_2(\text{atoms})$$

- input word represents an undirected graph
- can nodes (atoms) be computed?

$$\{a, b\} \mapsto a$$

$$(\{a, b\}, \{b, c\}) \mapsto b$$

- then any decidable property of undirected graphs can be decided deterministically

alphabet: unordered pairs of atoms

$\{a, b\} \in \mathcal{P}_2(\text{atoms})$

- input word represents an undirected graph
- can nodes (atoms) be computed?

$$\sout{\{a, b\} \mapsto a}$$

$$(\{a, b\}, \{b, c\}) \mapsto b$$

- then any decidable property of undirected graphs can be decided deterministically

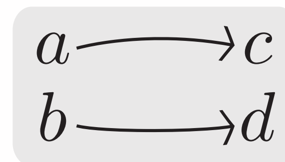
Fact: TMs over this alphabet do determinize

alphabet: unordered pairs of ordered pairs of atoms

$$\{(a,c),(b,d)\}$$

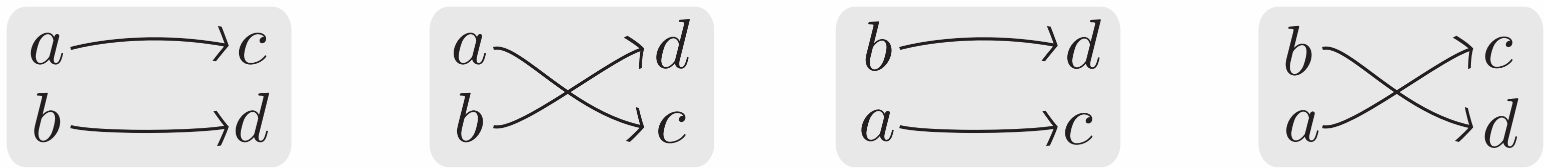
alphabet: unordered pairs of ordered pairs of atoms

$$\{(a, c), (b, d)\}$$



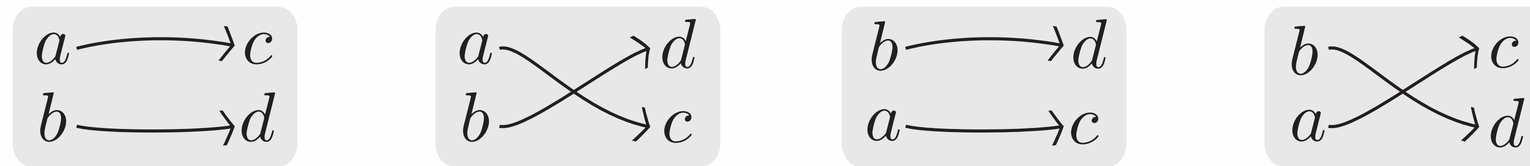
alphabet: unordered pairs of ordered pairs of atoms

$$\{(a,c),(b,d)\}$$

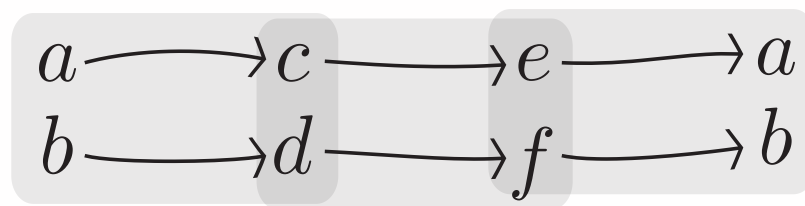


alphabet: unordered pairs of ordered pairs of atoms

$$\{(a, c), (b, d)\}$$

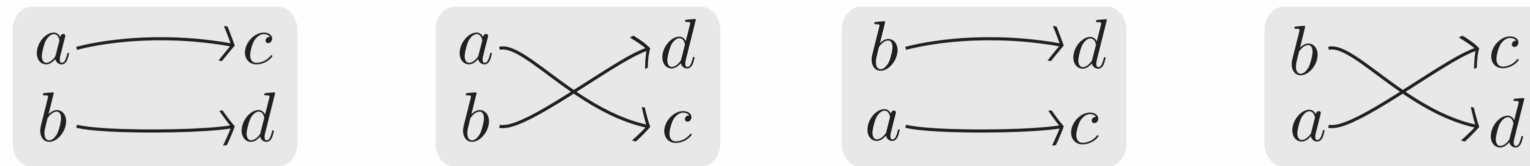


simple strips:

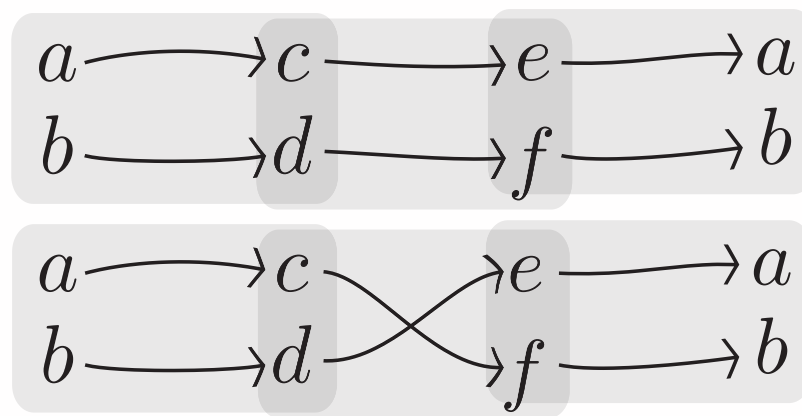


alphabet: unordered pairs of ordered pairs of atoms

$$\{(a, c), (b, d)\}$$



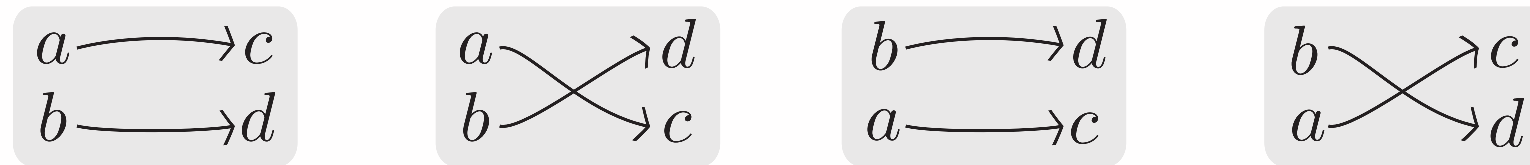
simple strips:



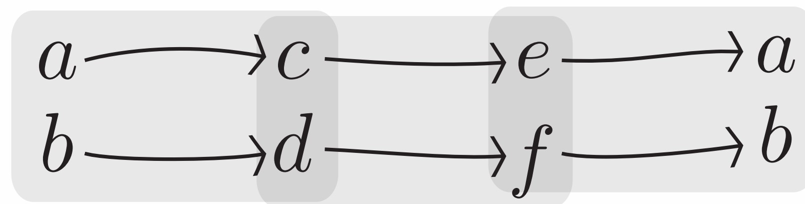
is not a simple strip

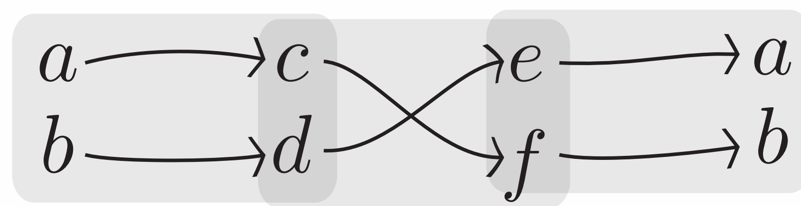
alphabet: unordered pairs of ordered pairs of atoms

$$\{(a, c), (b, d)\}$$

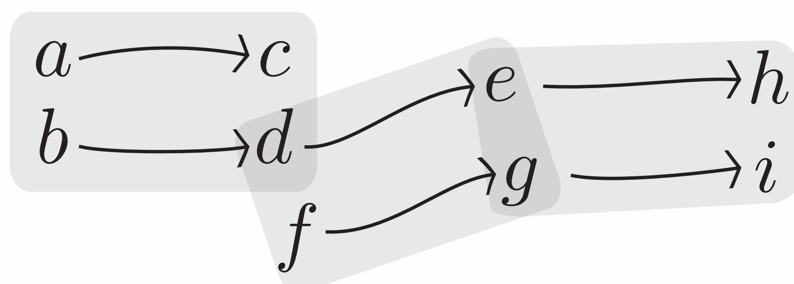


simple strips:





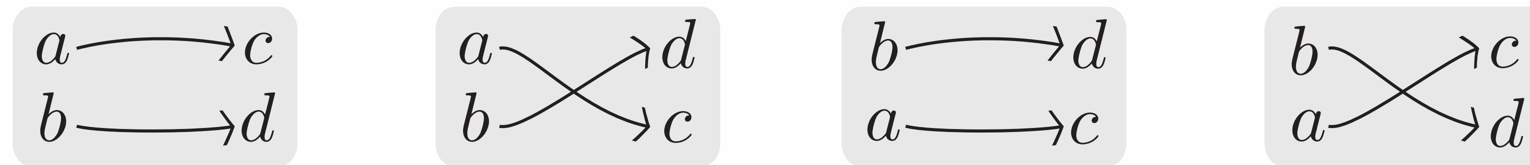
is not a simple strip



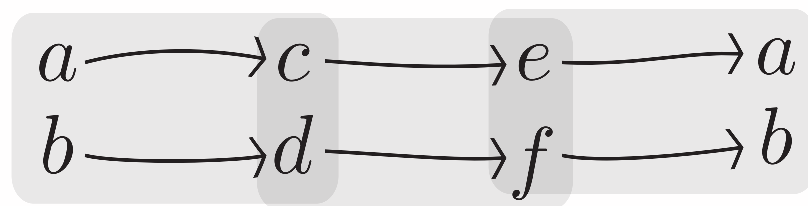
neither

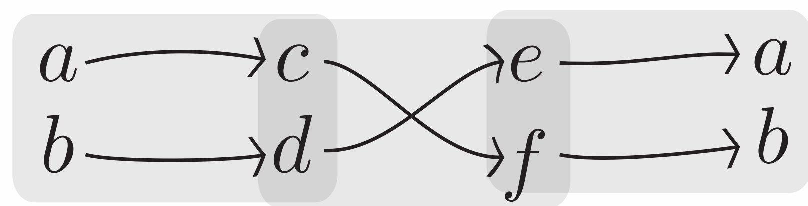
alphabet: unordered pairs of ordered pairs of atoms

$$\{(a,c),(b,d)\}$$

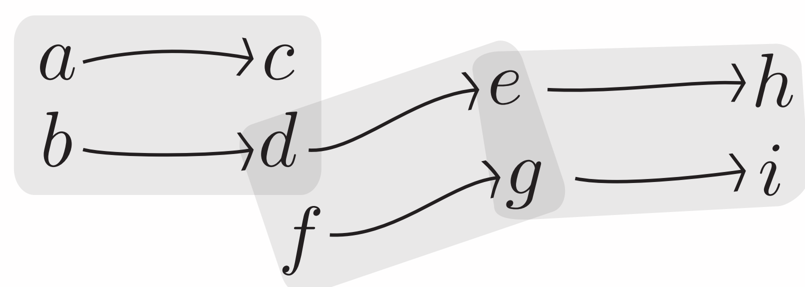


simple strips:





is not a simple strip

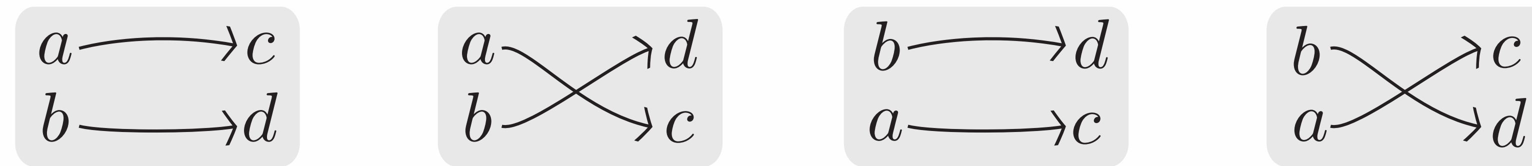


neither

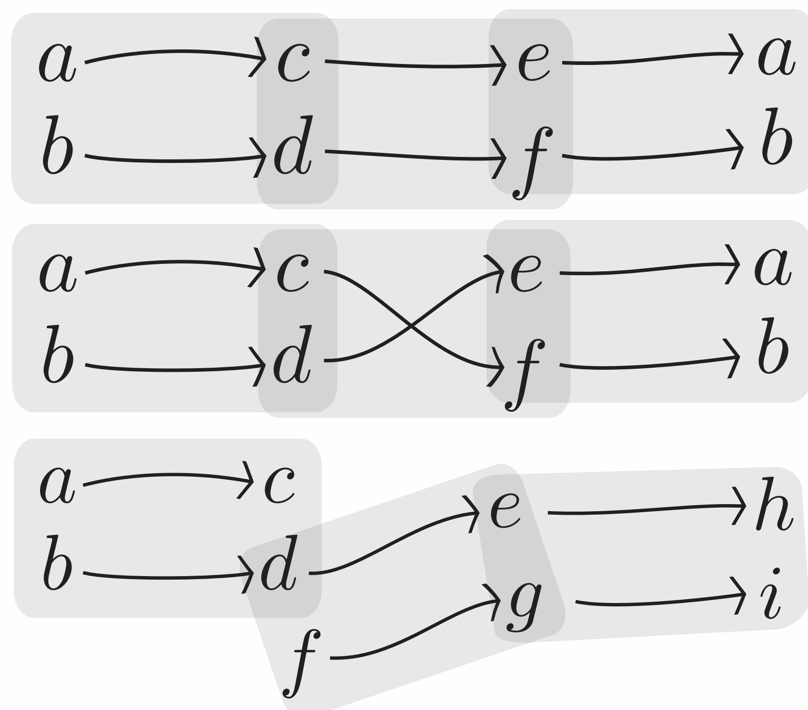
Are simple strips recognized by a nondeterministic TM?

alphabet: unordered pairs of ordered pairs of atoms

$$\{(a,c),(b,d)\}$$



simple strips:



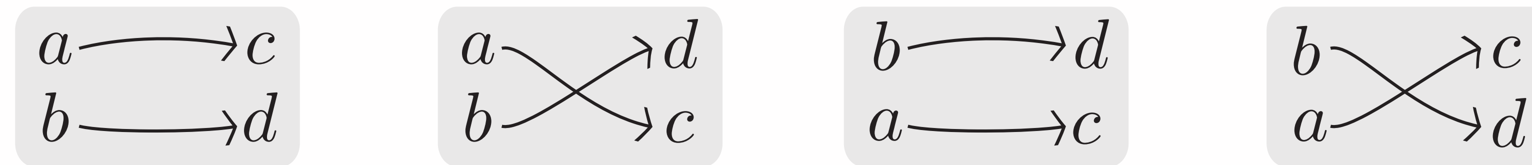
is not a simple strip

neither

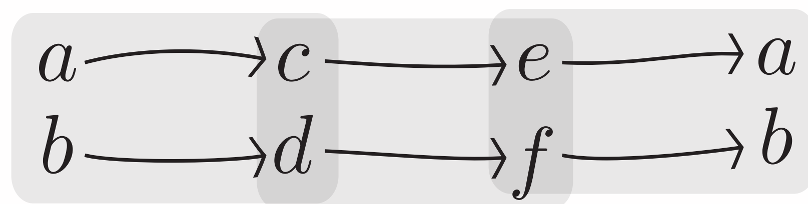
Are simple strips recognized by a deterministic TM?

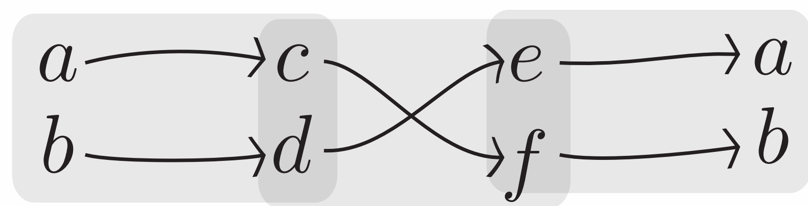
alphabet: unordered pairs of ordered pairs of atoms

$$\{(a, c), (b, d)\}$$

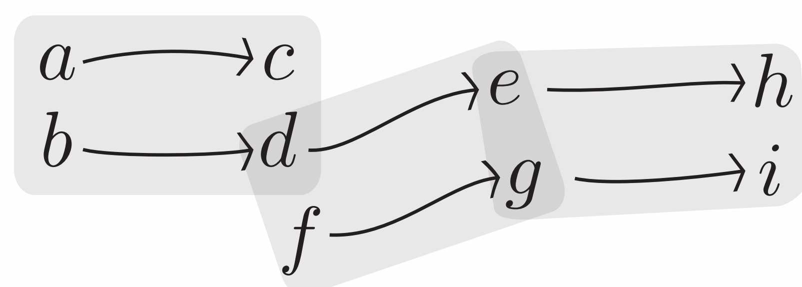


simple strips:



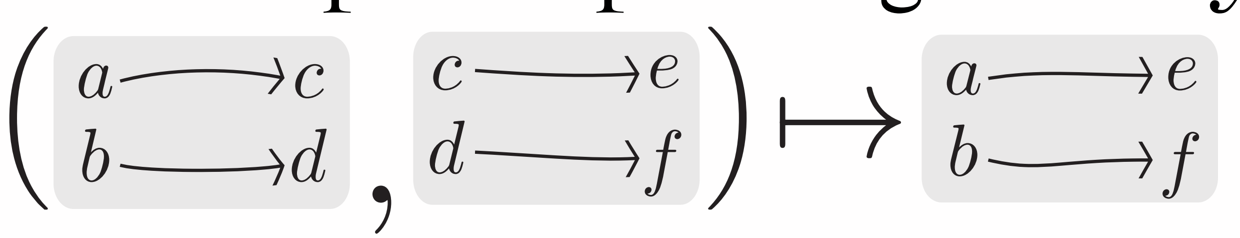


is not a simple strip



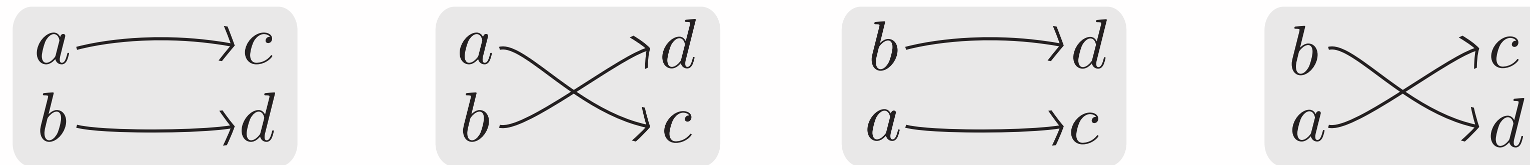
neither

Are simple strips recognized by a deterministic TM?

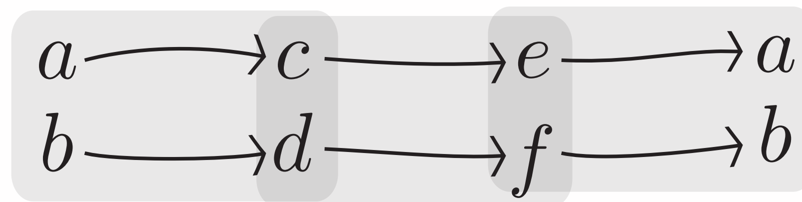


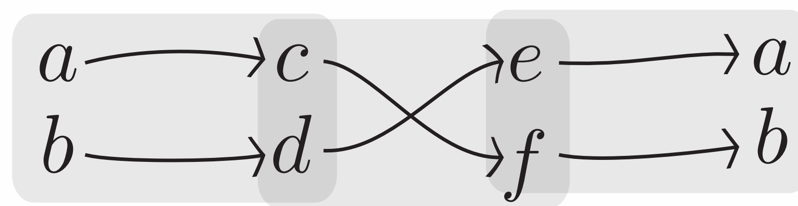
alphabet: unordered pairs of ordered pairs of atoms

$\{(a,c),(b,d)\}$

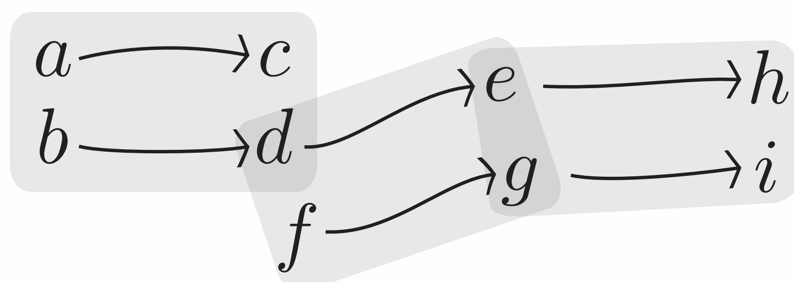


simple strips:



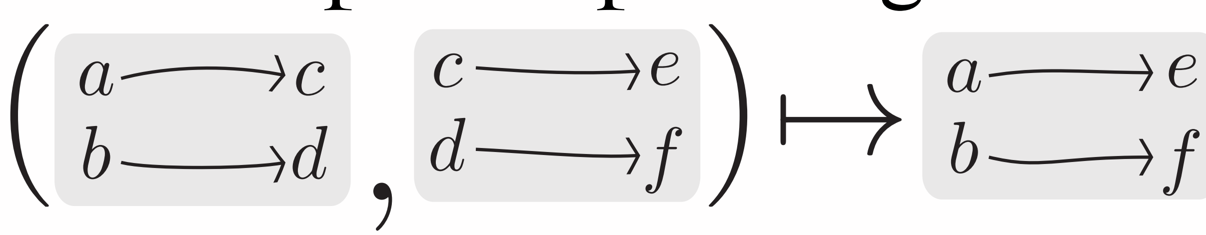


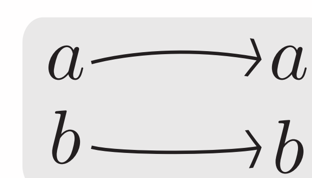
is not a simple strip



neither

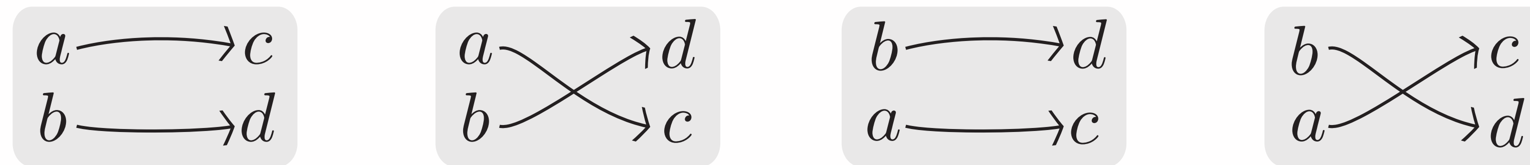
Are simple strips recognized by a deterministic TM?



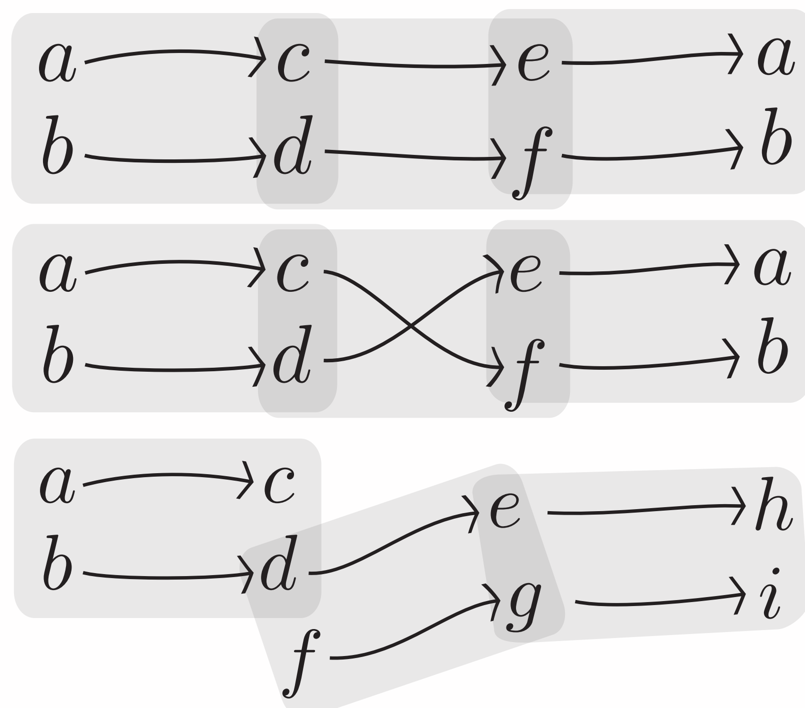


alphabet: unordered pairs of ordered pairs of atoms

$$\{(a,c),(b,d)\}$$



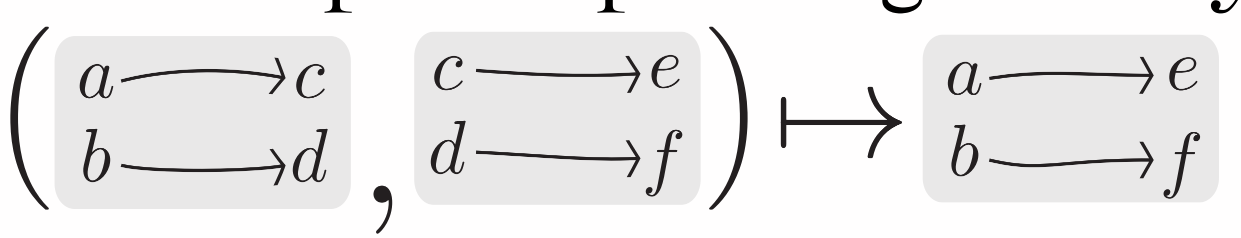
simple strips:



is not a simple strip

neither

Are simple strips recognized by a deterministic TM?



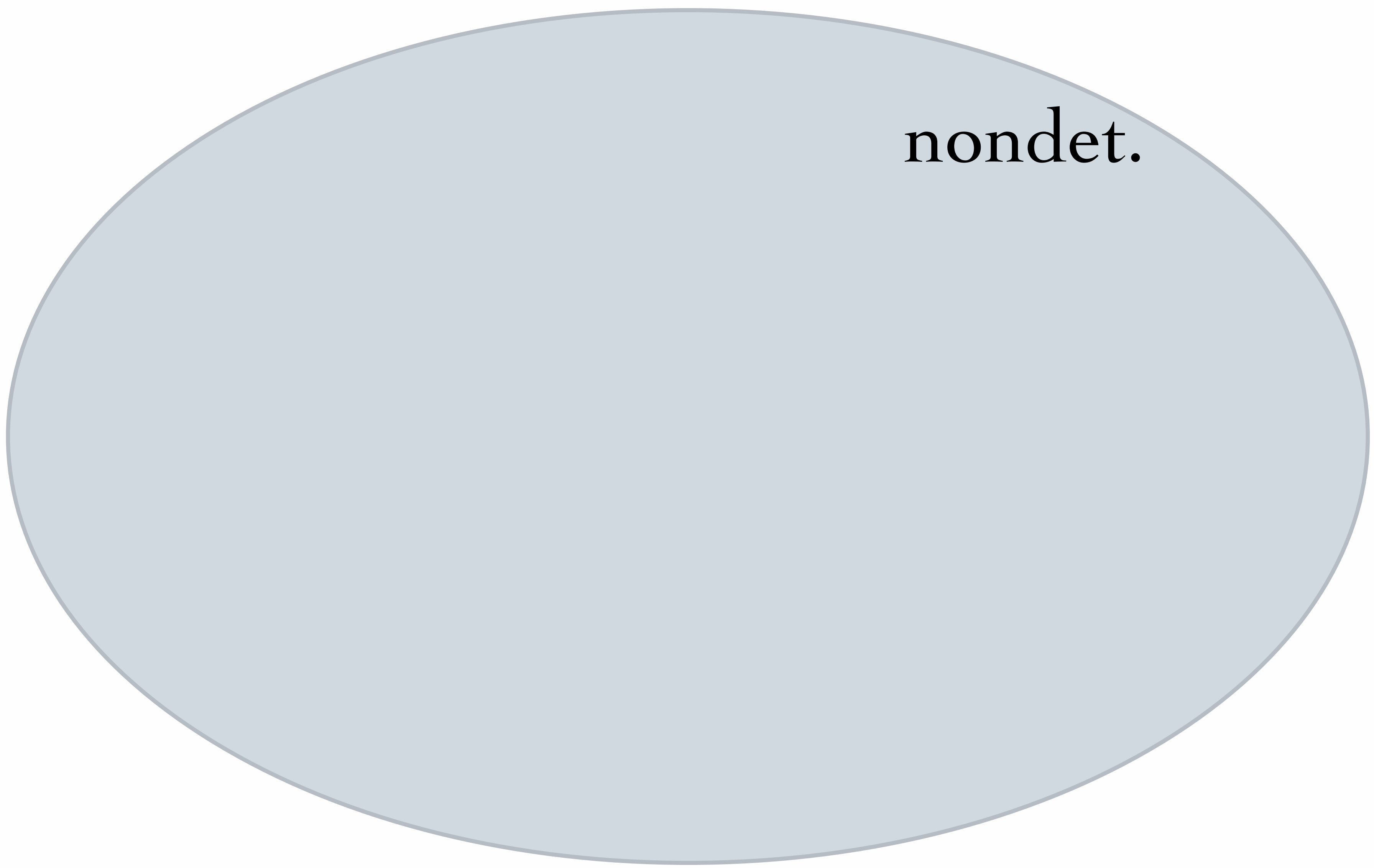
Fact: TMs over this alphabet do determinize

Theorem:

There is an alphabet A, and a language over A that is in NP but is not recognizable by a deterministic TM.

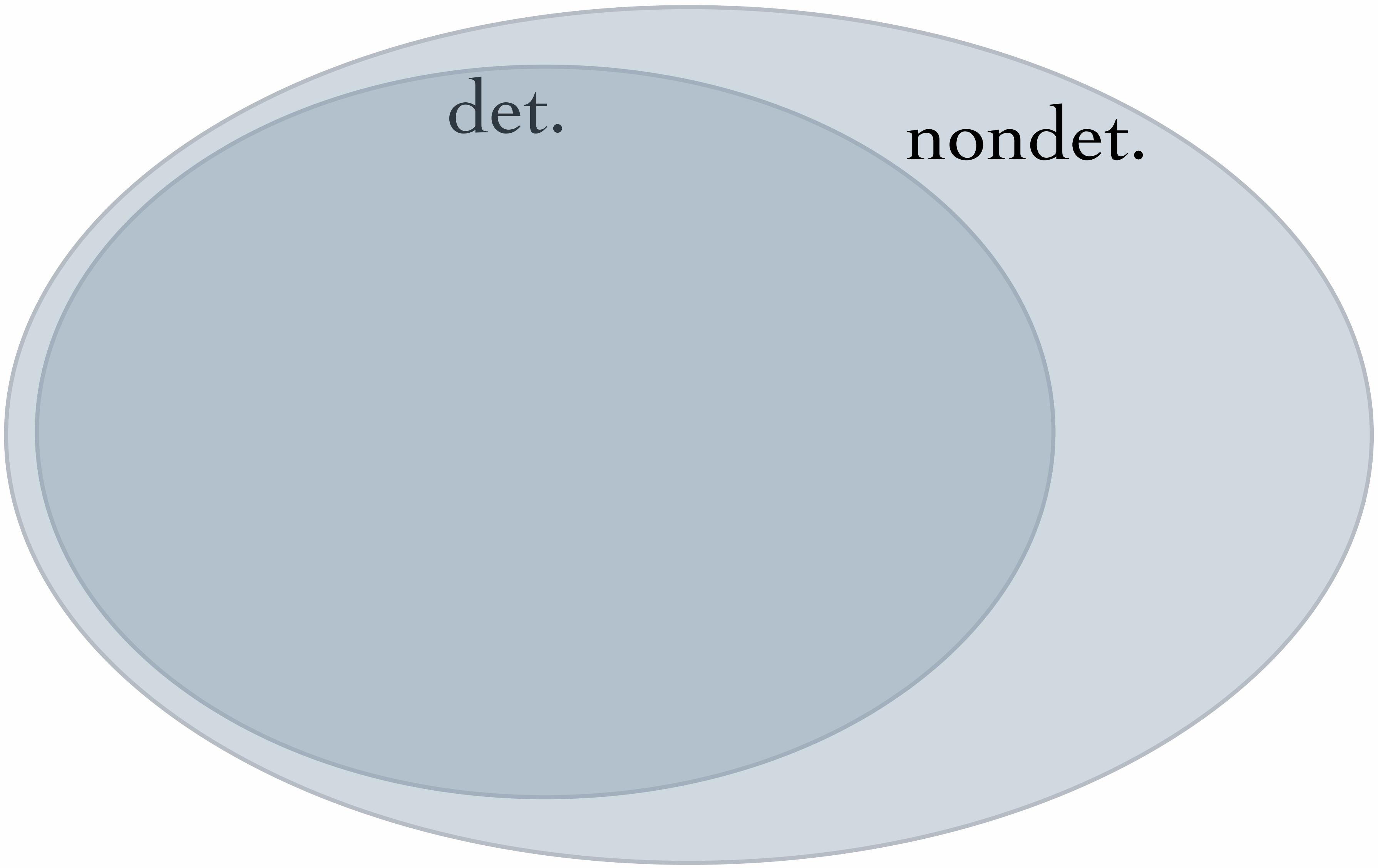
Theorem:

There is an alphabet A, and a language over A that is in NP but is not recognizable by a deterministic TM.



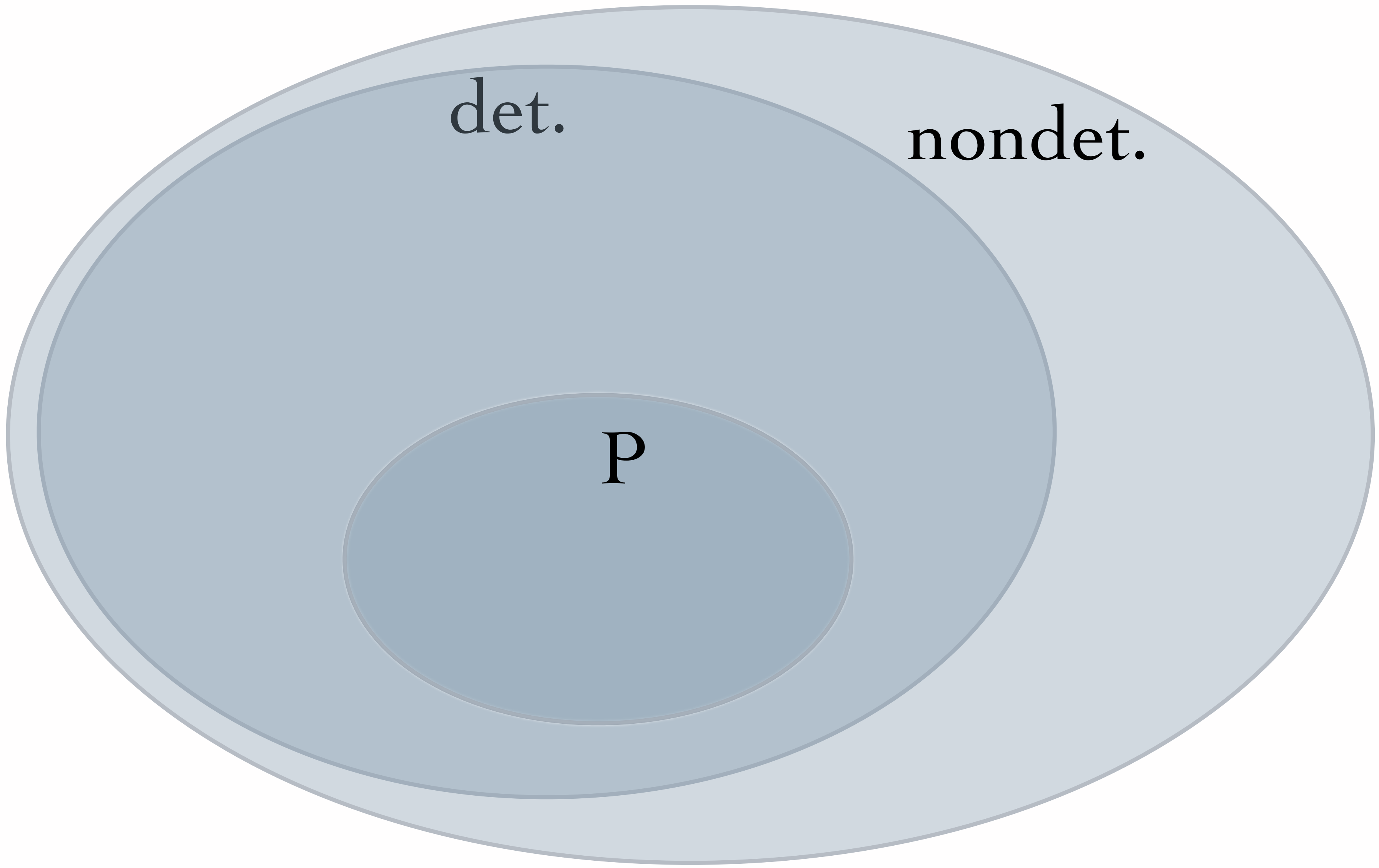
Theorem:

There is an alphabet A, and a language over A that is in NP but is not recognizable by a deterministic TM.



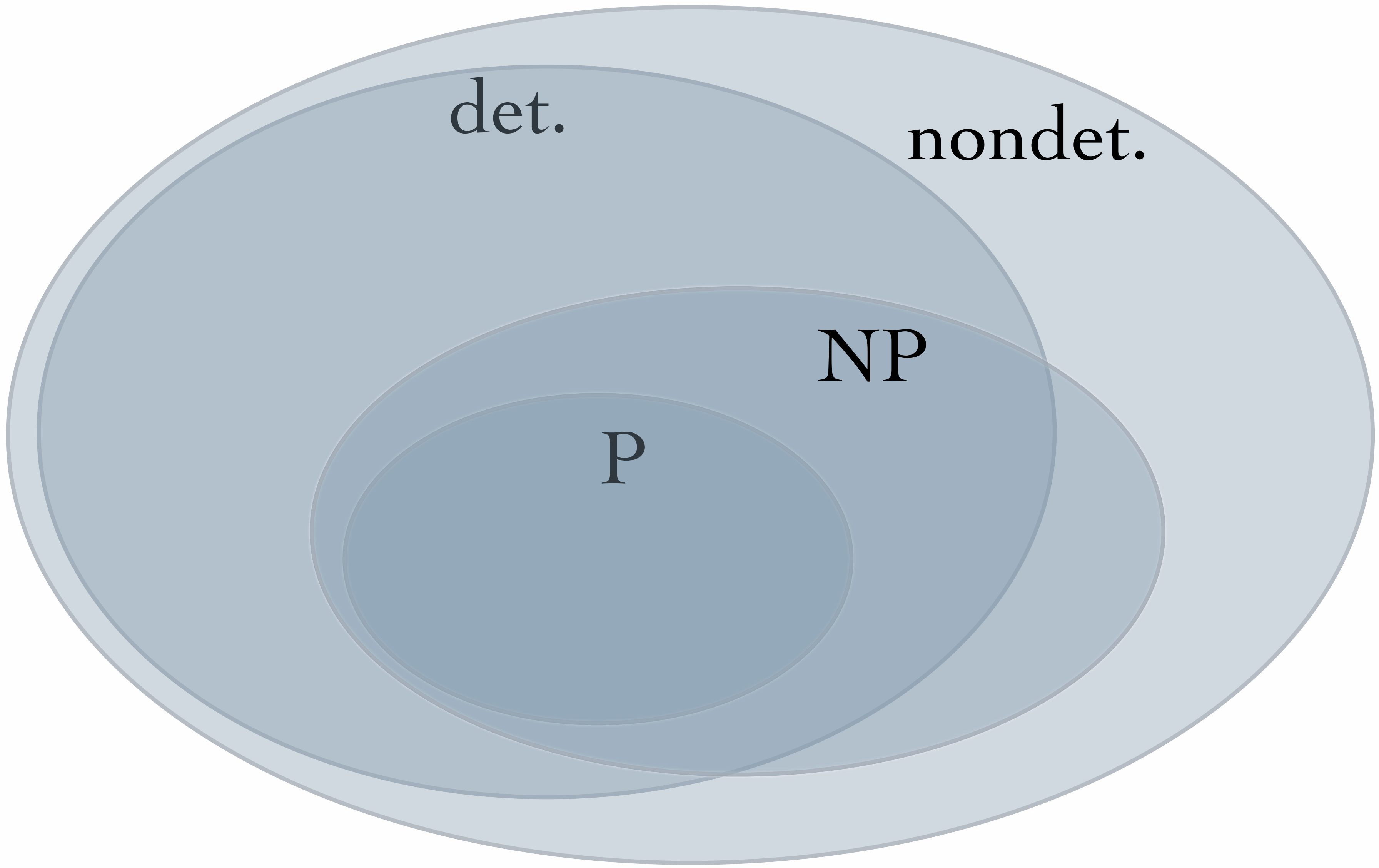
Theorem:

There is an alphabet A, and a language over A that is in NP but is not recognizable by a deterministic TM.



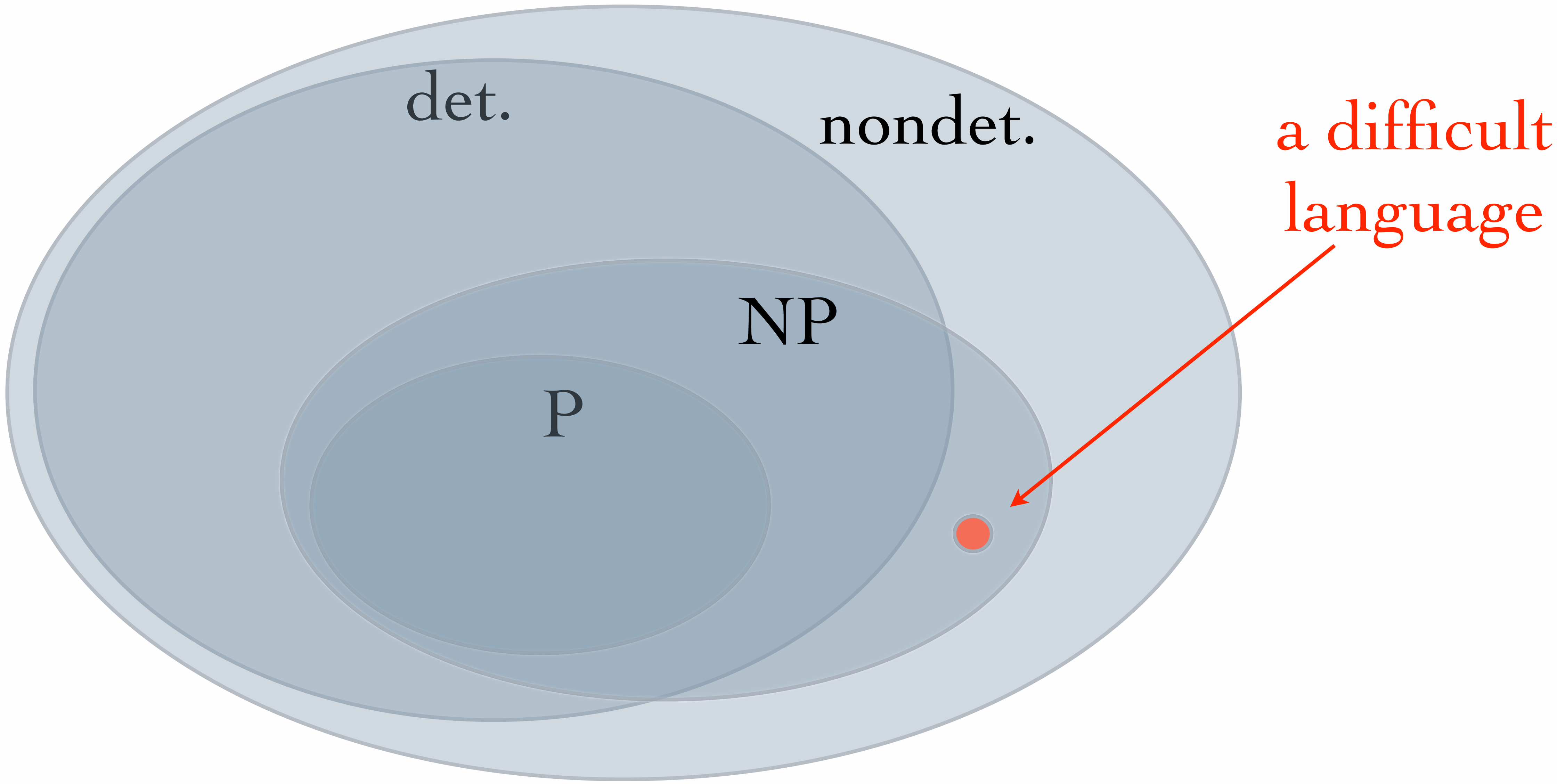
Theorem:

There is an alphabet A, and a language over A that is in NP but is not recognizable by a deterministic TM.



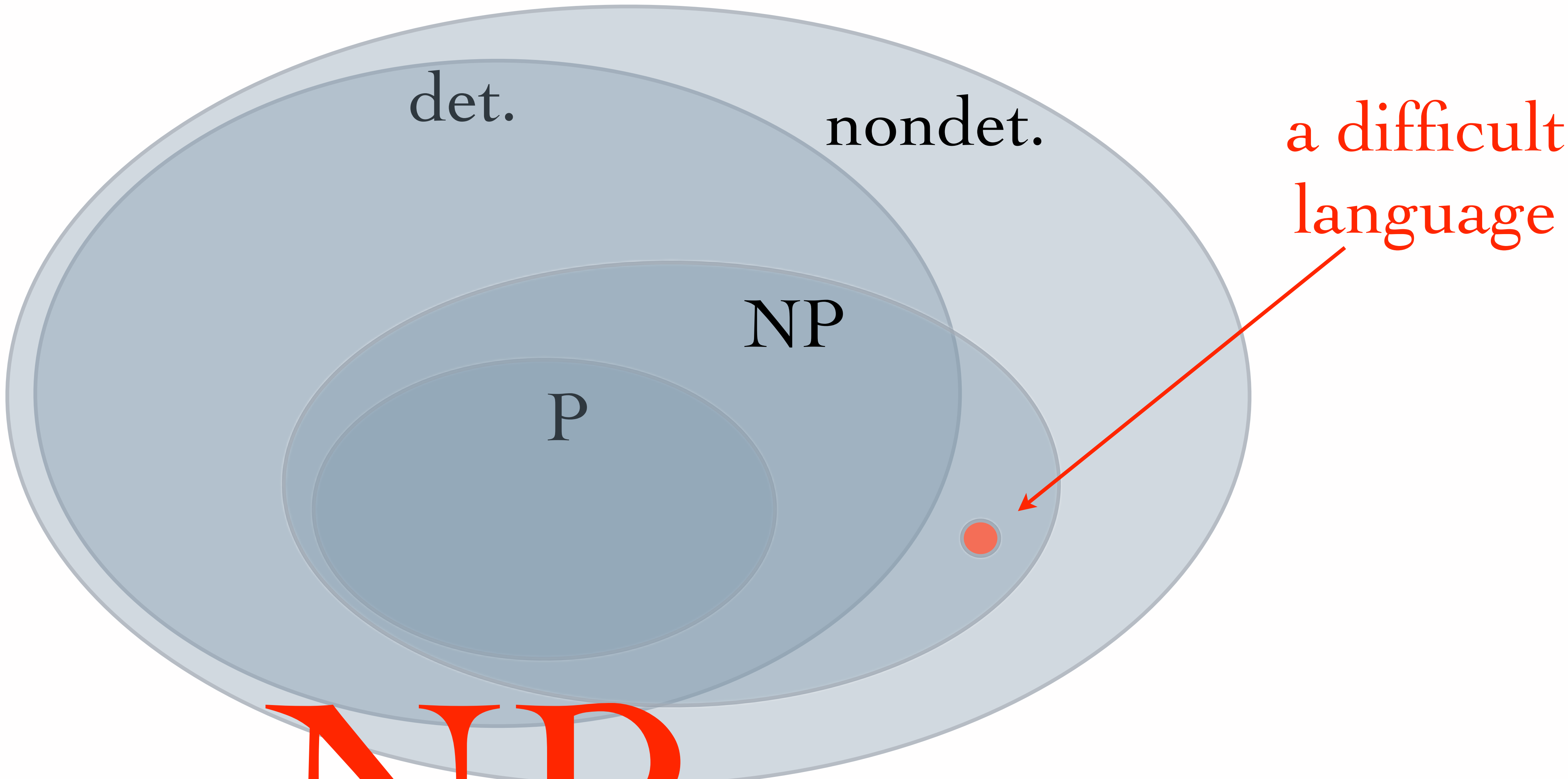
Theorem:

There is an alphabet A, and a language over A that is in NP but is not recognizable by a deterministic TM.



Theorem:

There is an alphabet A, and a language over A that is in NP but is not recognizable by a deterministic TM.

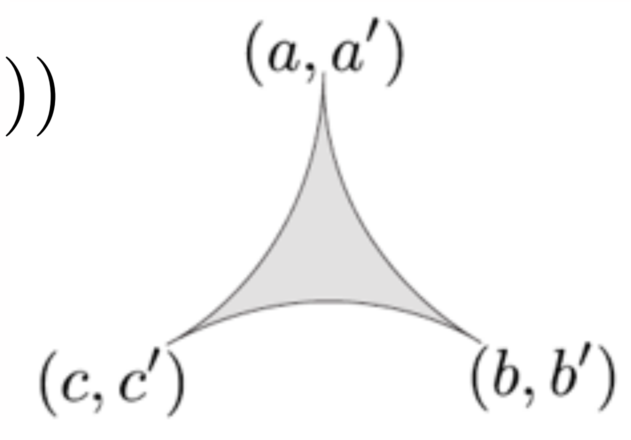


P ≠ NP

alphabet: ordered triples of ordered pairs of atoms modulo even number of flips

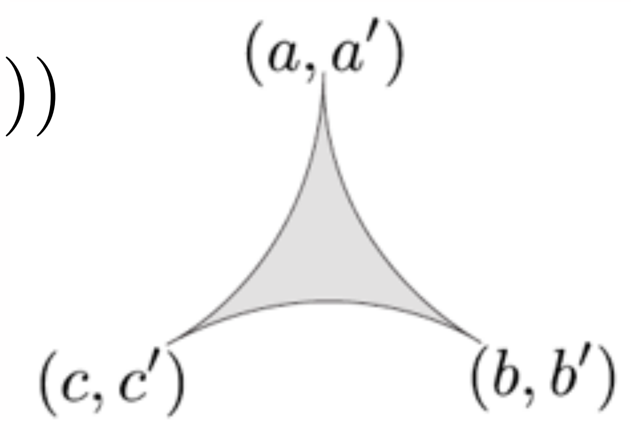
alphabet: ordered triples of ordered pairs of atoms modulo even number of flips

Triangle = $((a, a'), (b, b'), (c, c'))$



alphabet: ordered triples of ordered pairs of atoms modulo even number of flips

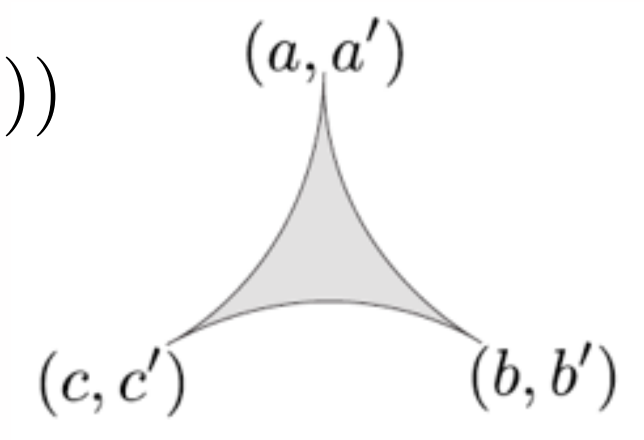
Triangle = $((a, a'), (b, b'), (c, c'))$



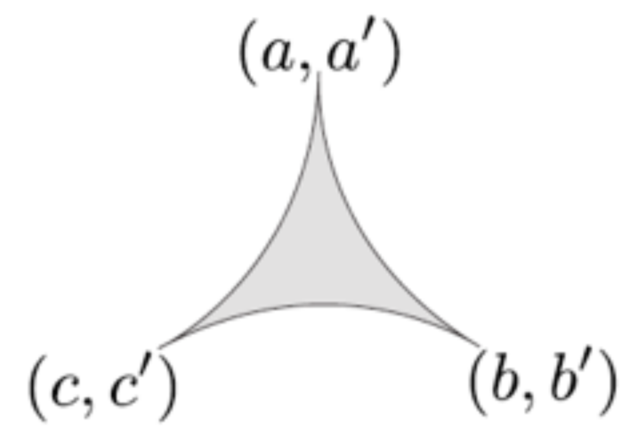
Let triangles with same pairs be equivalent if exactly two pairs are flipped:

alphabet: ordered triples of ordered pairs of atoms modulo even number of flips

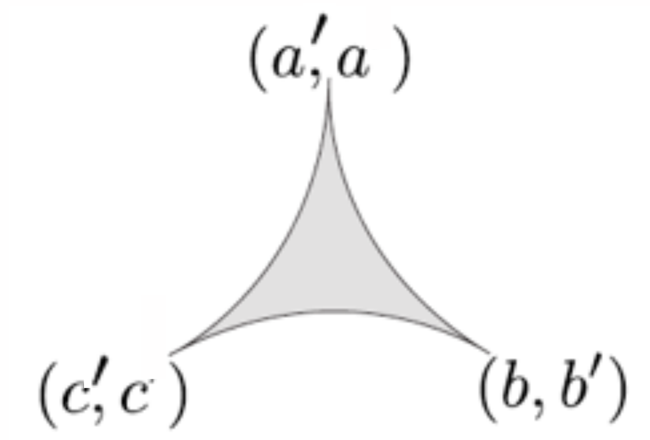
Triangle = $((a, a'), (b, b'), (c, c'))$



Let triangles with same pairs be equivalent if exactly two pairs are flipped:

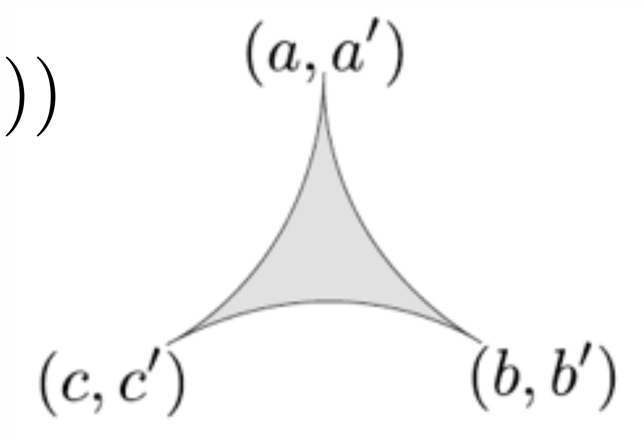


$\approx$

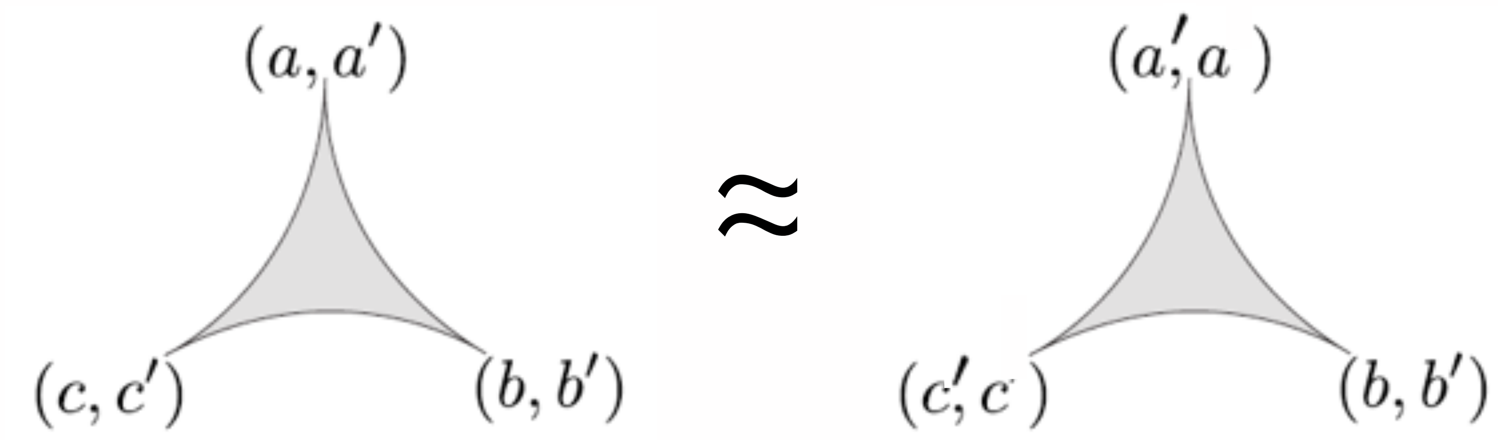


alphabet: ordered triples of ordered pairs of atoms modulo even number of flips

Triangle = $((a, a'), (b, b'), (c, c'))$

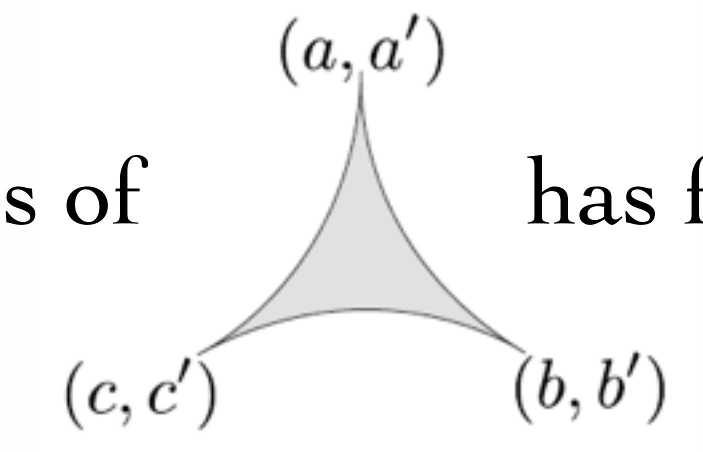


Let triangles with same pairs be equivalent if exactly two pairs are flipped:



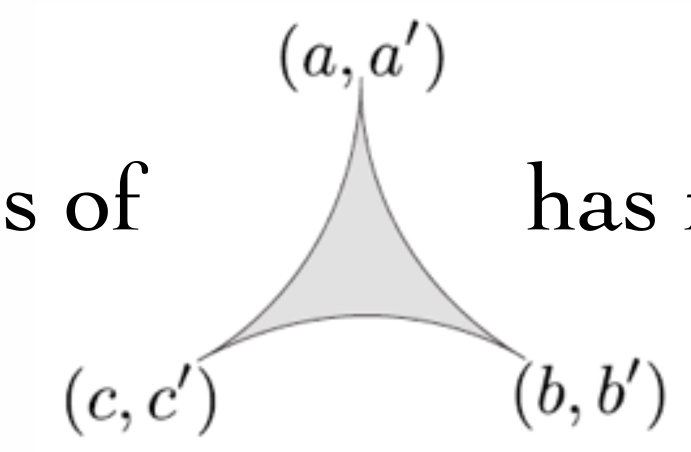
Alphabet = equivalence classes of triangles

alphabet: ordered triples of ordered pairs of atoms modulo even number of flips

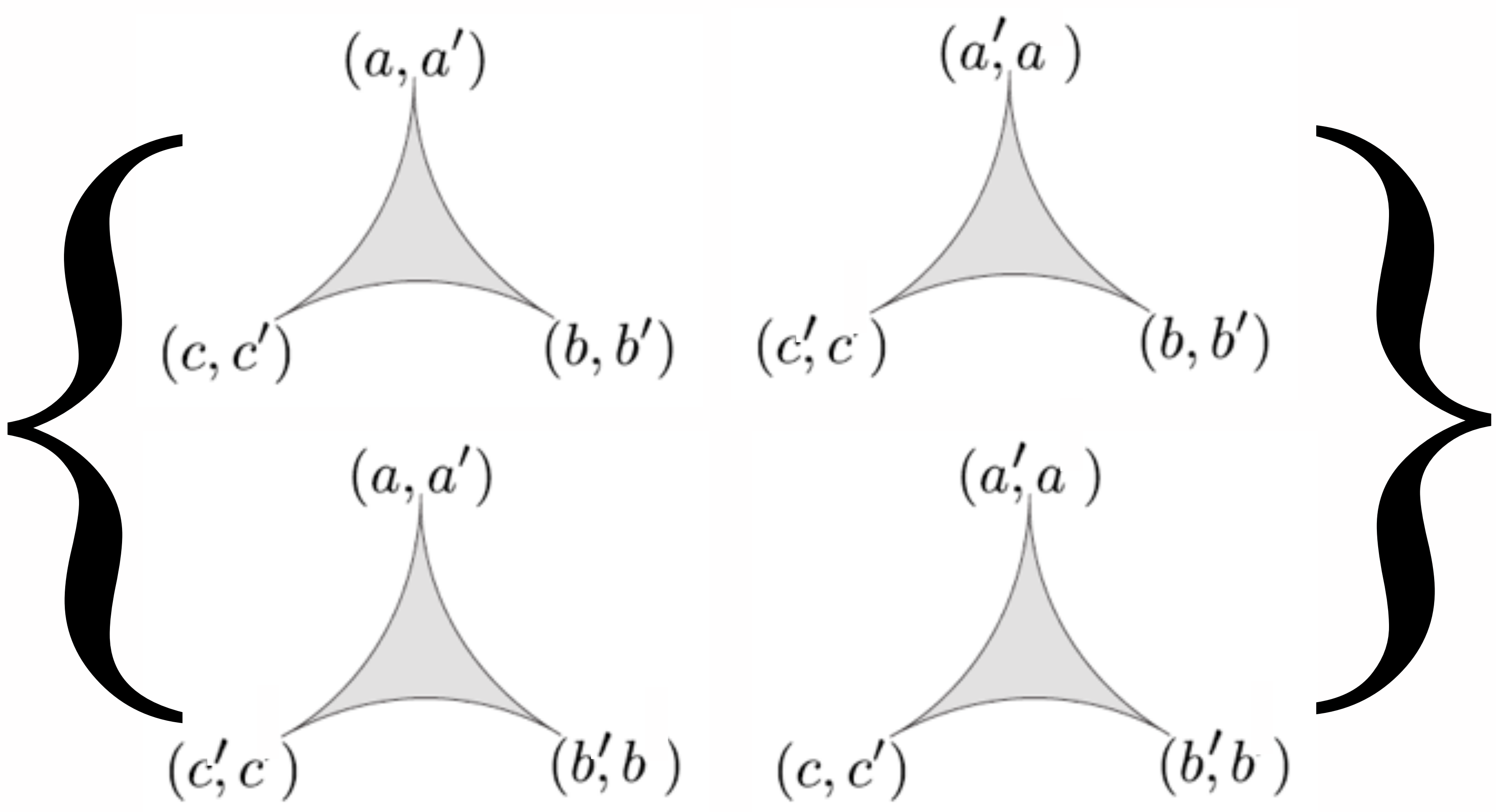
equivalence class of  has four elements:

alphabet: ordered triples of ordered pairs of atoms modulo even number of flips

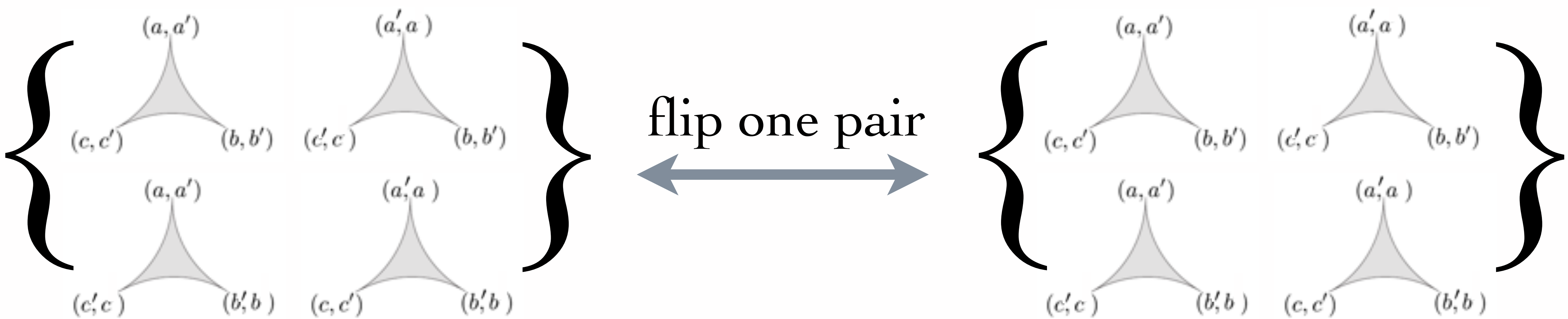
equivalence class of



has four elements:

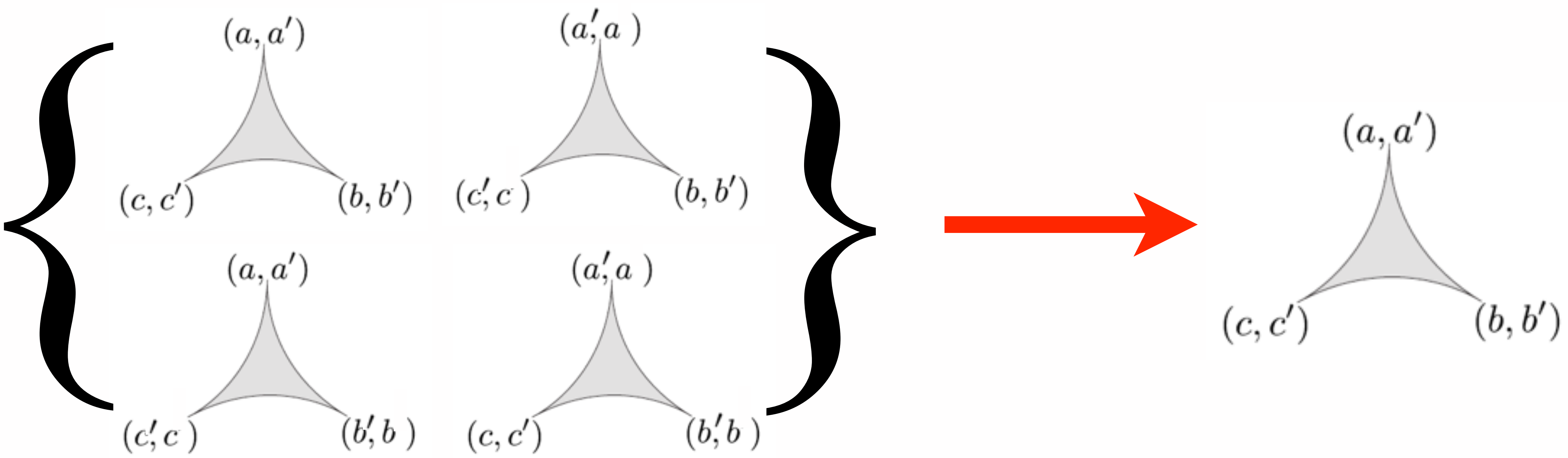


alphabet: ordered triples of ordered pairs of atoms modulo even number of flips

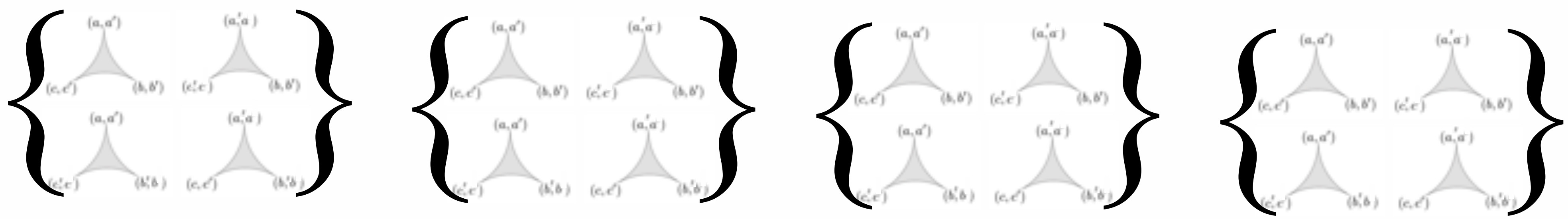


alphabet: ordered triples of ordered pairs of atoms modulo even number of flips

there is no function!



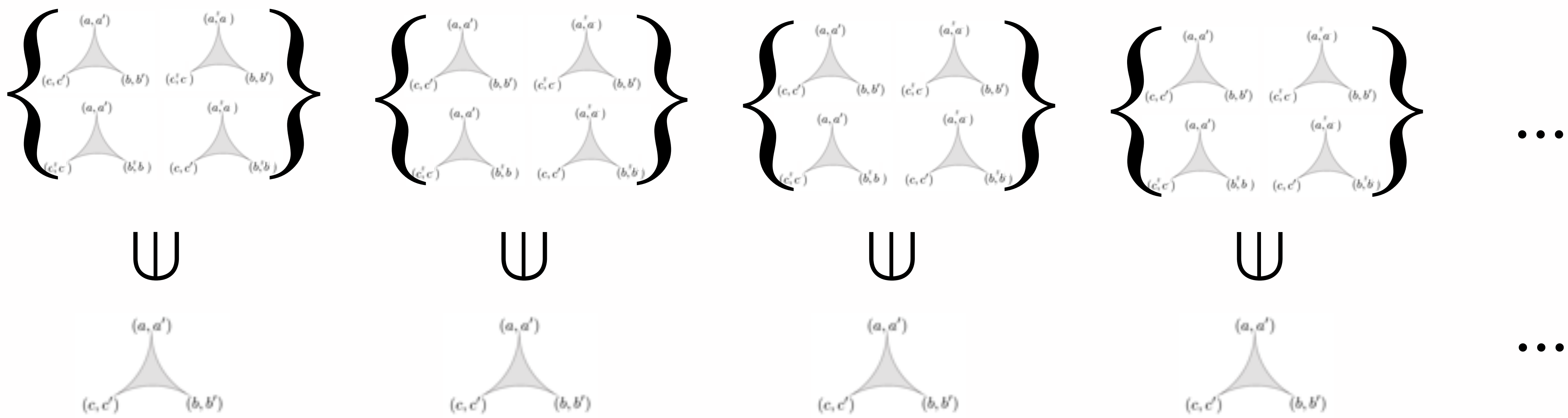
difficult language



...

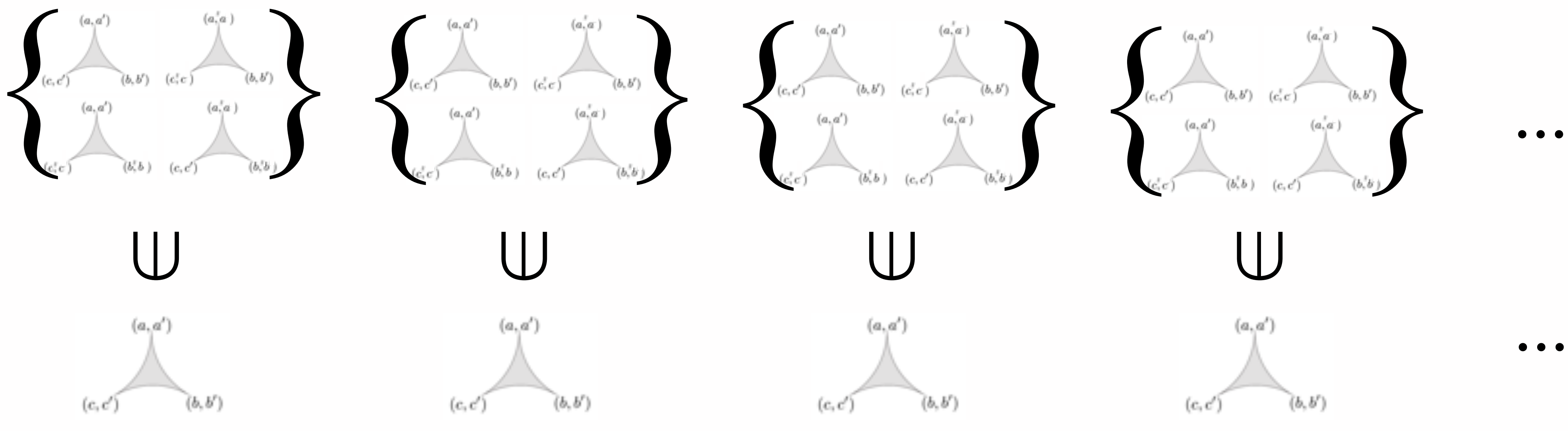
difficult language

sequence of elements



difficult language

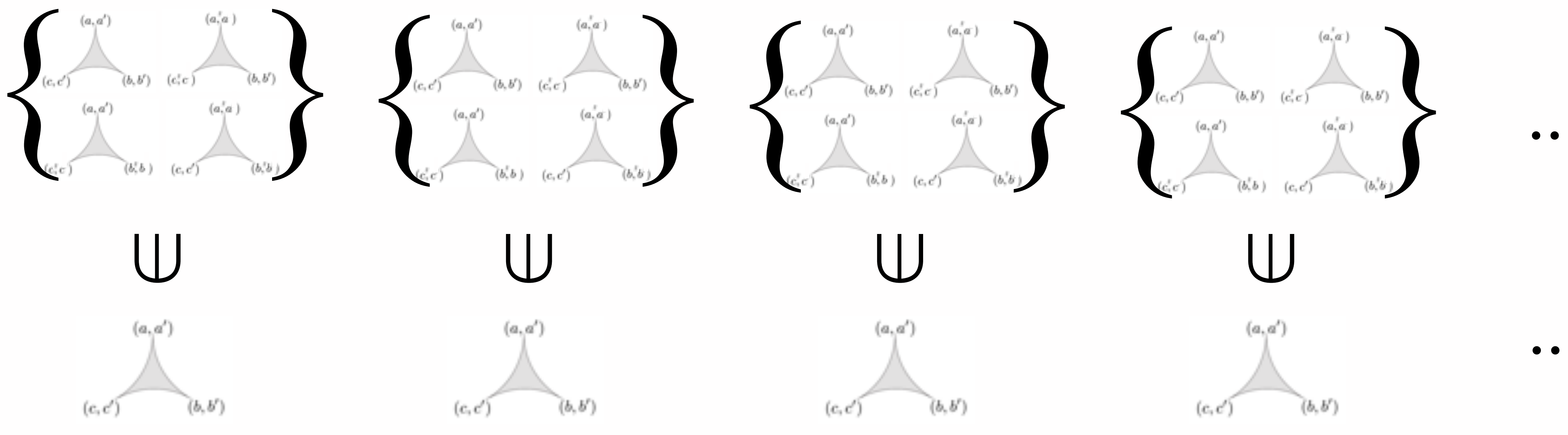
sequence
of
elements



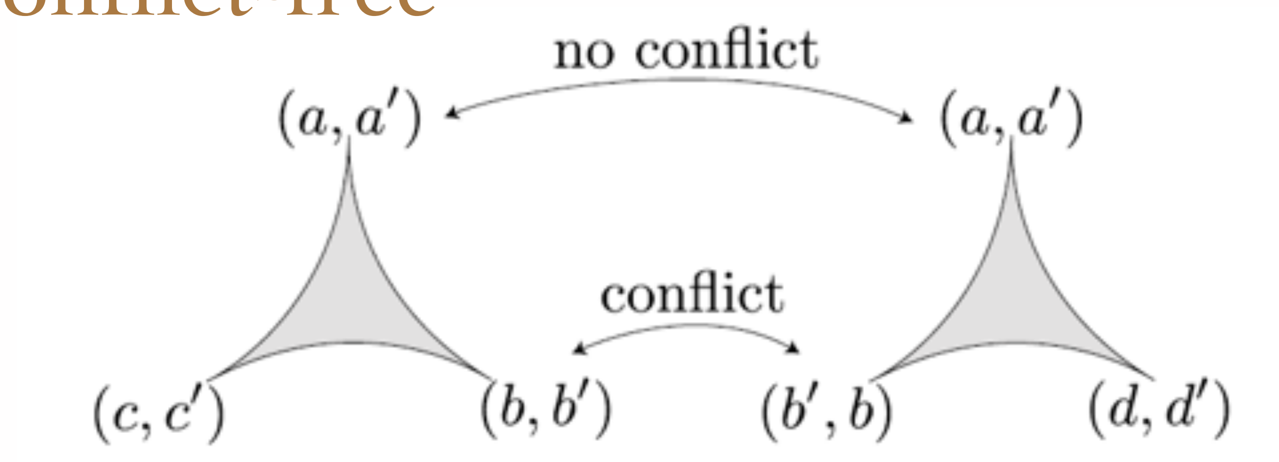
Language: a word is in the language iff
some sequence of elements is conflict-free

difficult language

sequence of elements

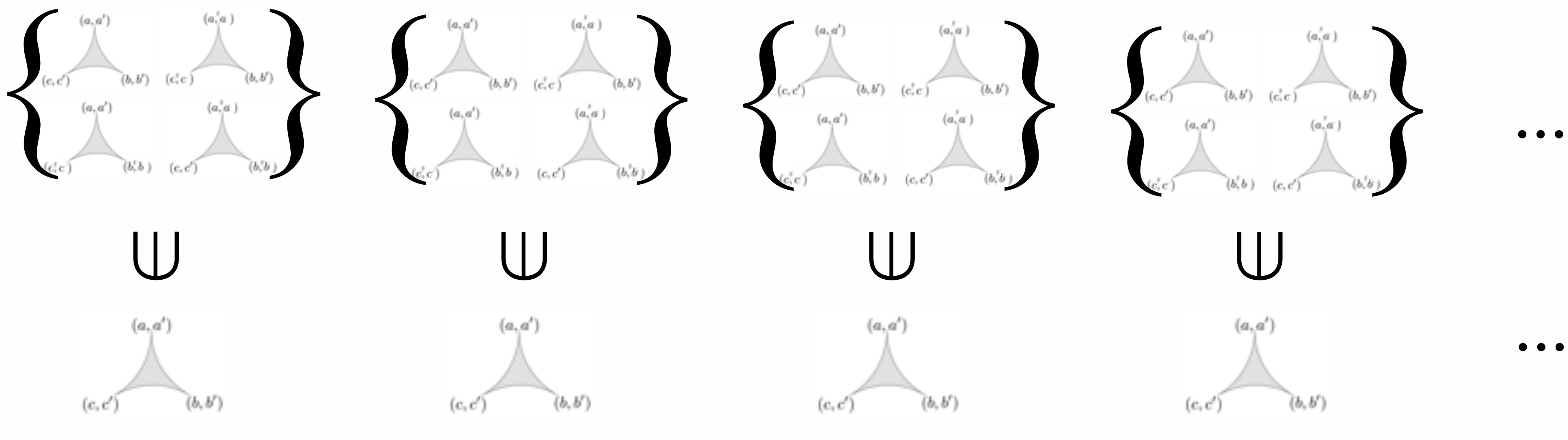


Language: a word is in the language iff some sequence of elements is conflict-free



difficult language

sequence of elements

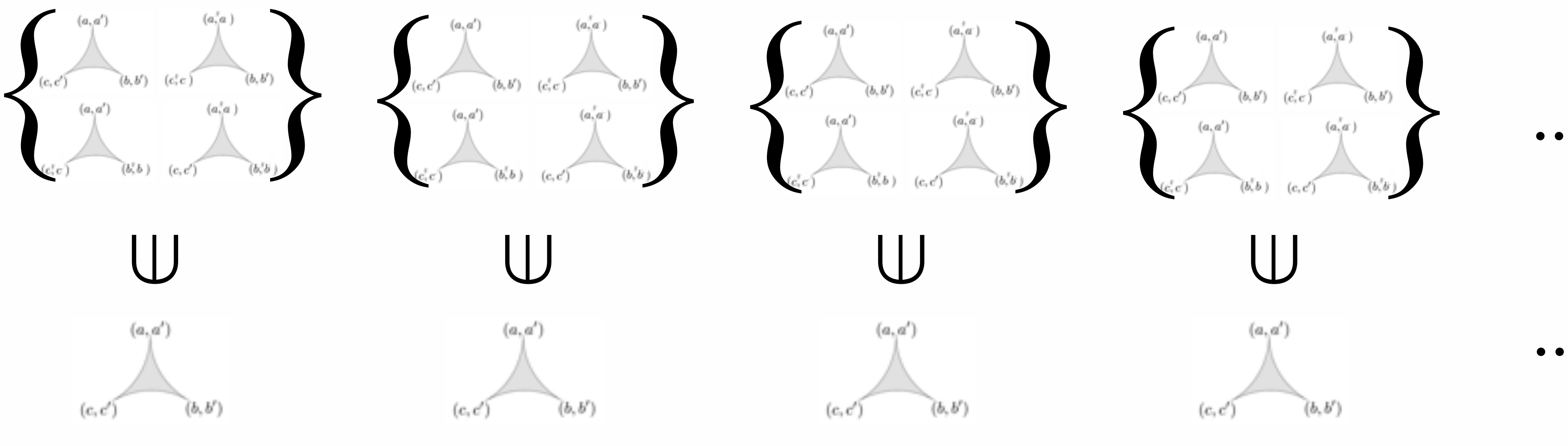


Language: a word is in the language iff some sequence of elements is conflict-free

closely related to Cai-Fuerer-Immerman graphs (1992)

difficult language

sequence of elements

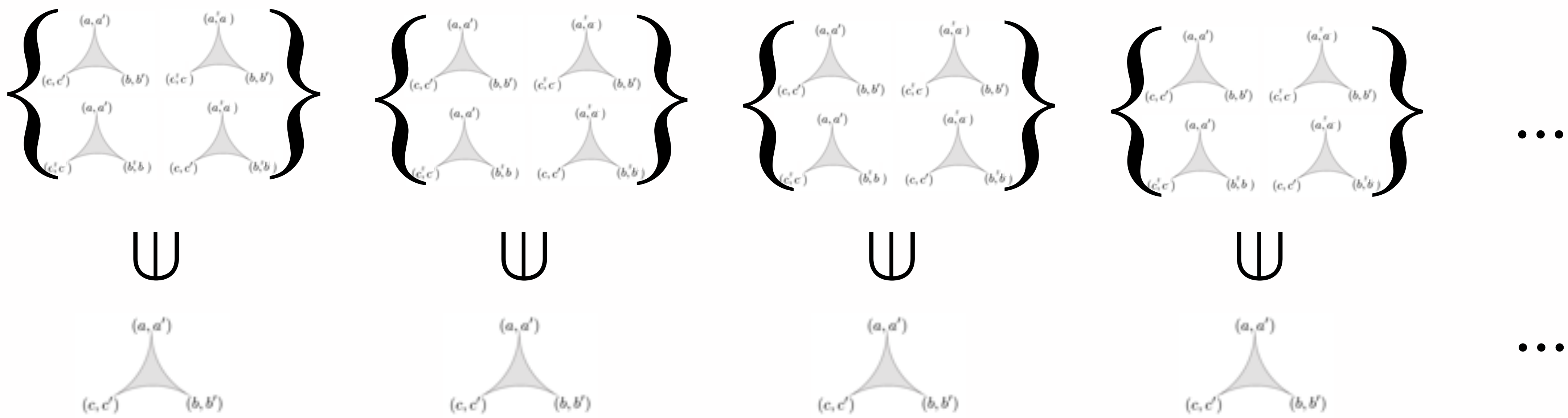


Language: a word is in the language iff some sequence of elements is conflict-free

recognized in NP: guess a sequence of elements

difficult language

sequence of elements

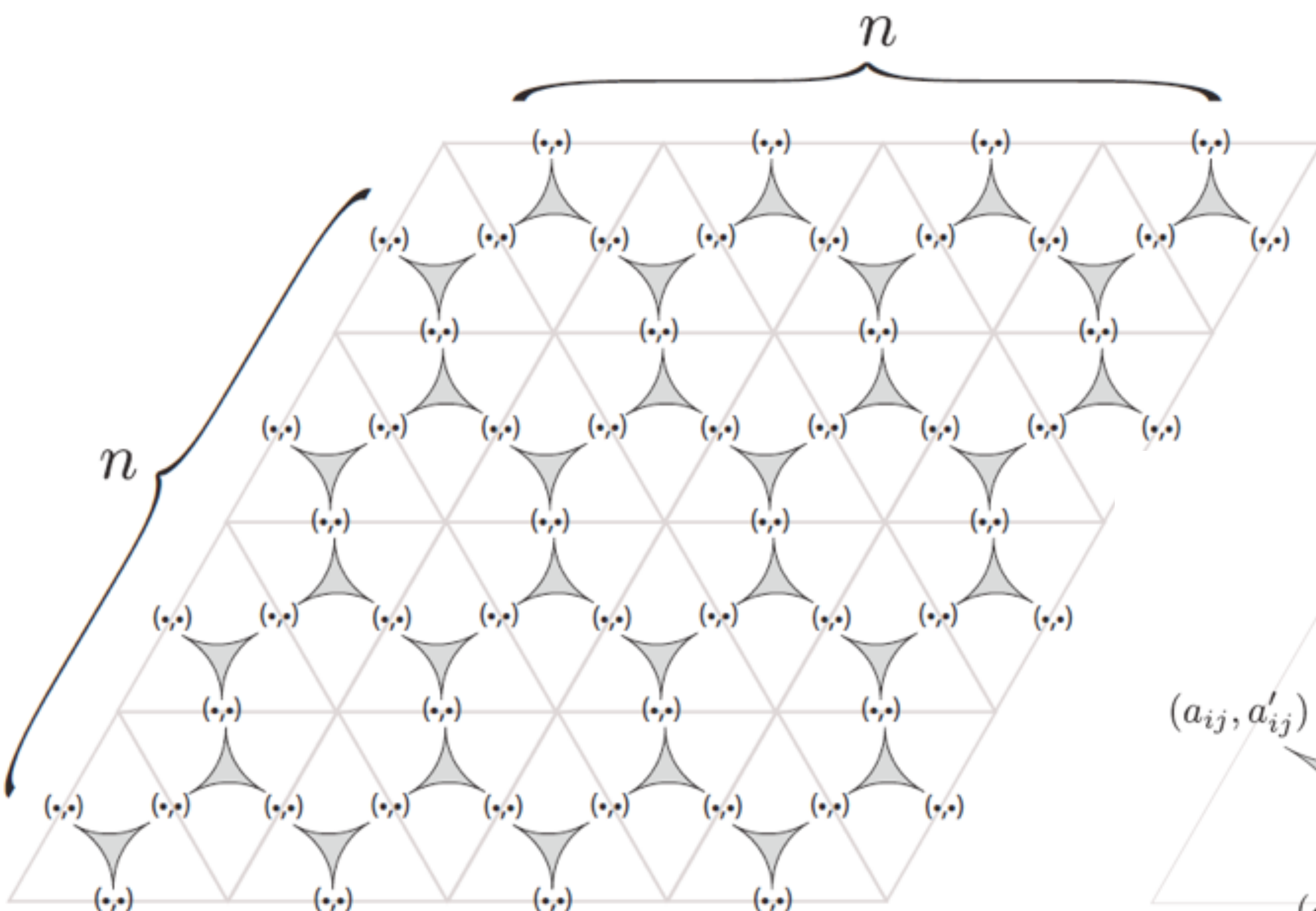


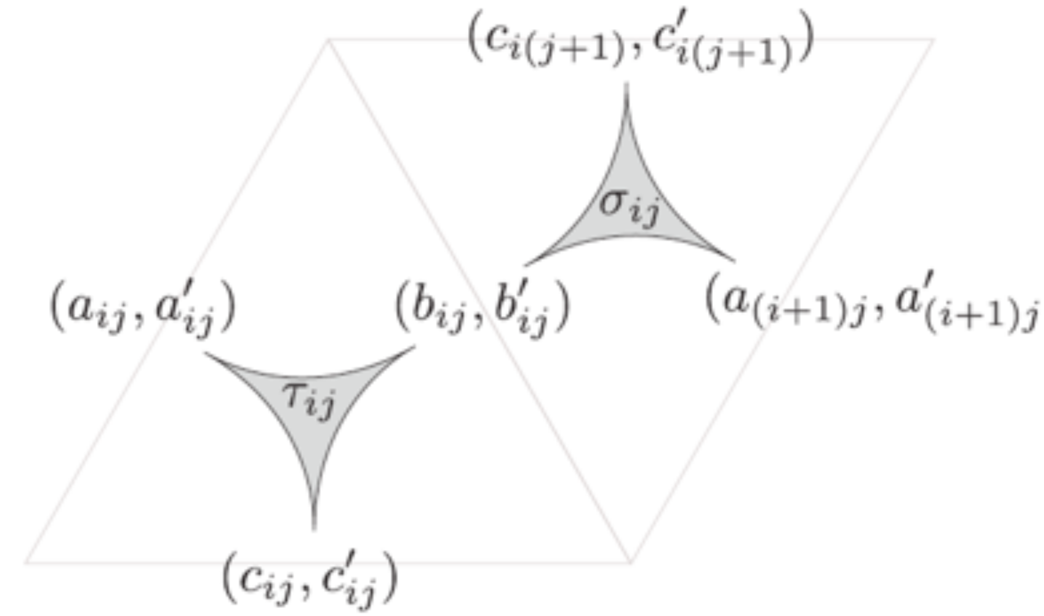
Language: a word is in the language iff some sequence of elements is conflict-free

recognized in NP: guess a sequence of elements

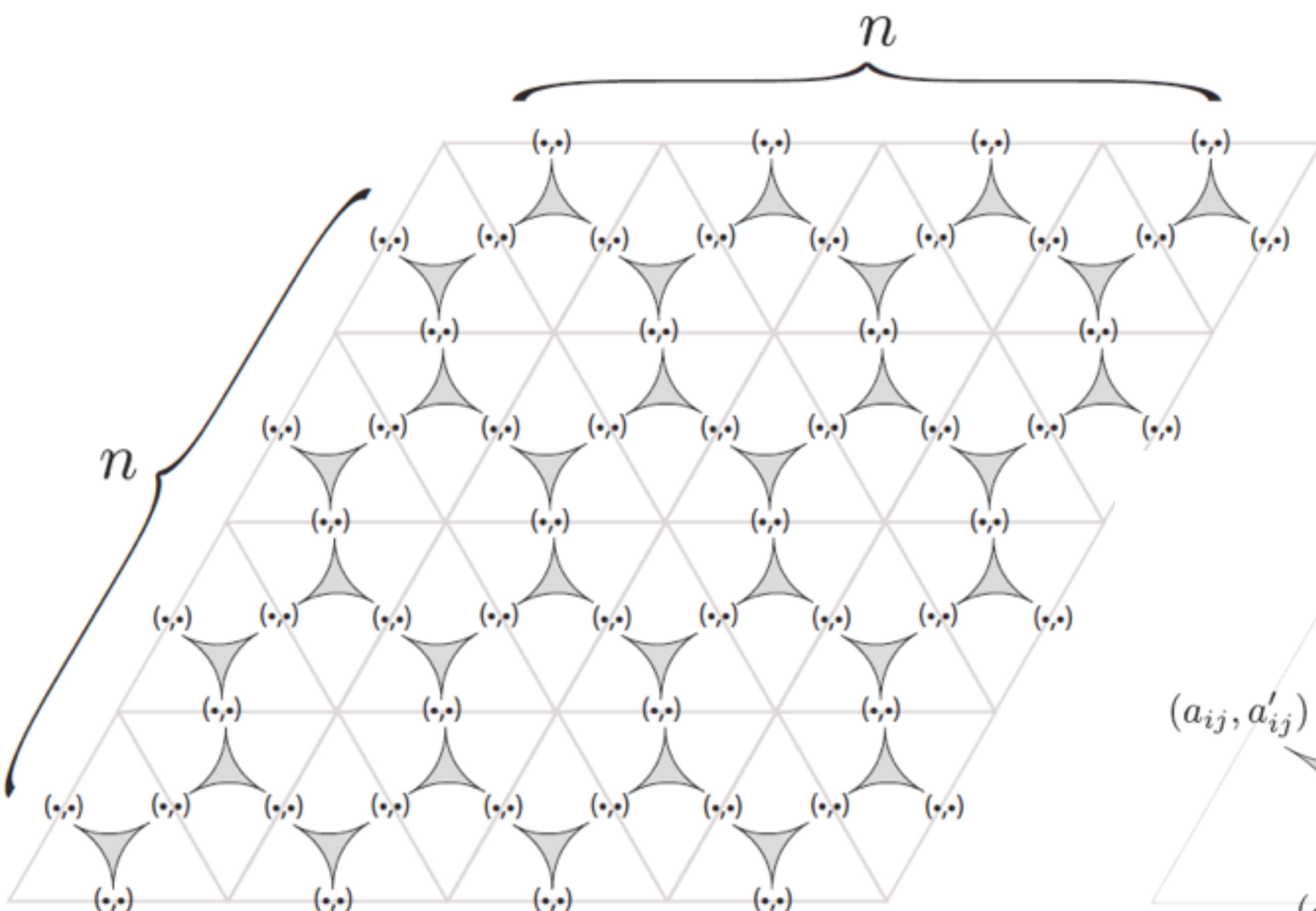
not recognized by a deterministic machine: enumeration of sequences of elements is not doable by a deterministic machine

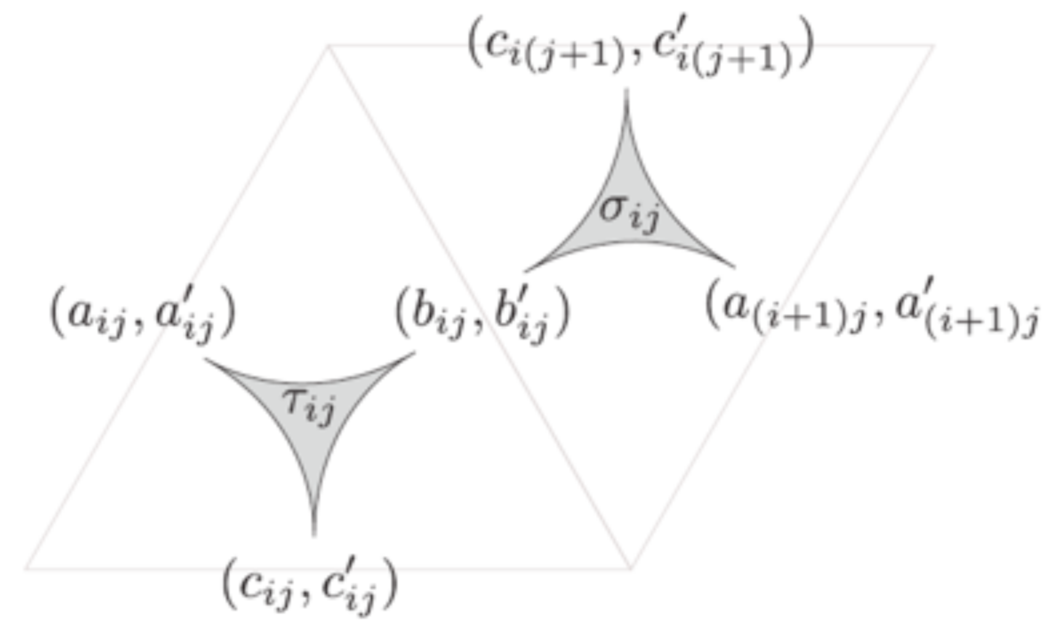
difficult language





difficult language





For sufficiently large size, deterministic machine can not distinguish an input torus from a "flipped" one

recent and on-going work

- effective characterization of standard alphabets using CSP theory
- model-theoretic characterization of standard alphabets (homogenizability)
- generalization to other well-behaved atoms
- applications to descriptive complexity
- characterization of standard atoms
- ...

recent and on-going work

- effective characterization of standard alphabets using CSP theory
- model-theoretic characterization of standard alphabets (homogenizability)
- generalization to other well-behaved atoms
- applications to descriptive complexity
- characterization of standard atoms
- ...

many interesting links to other fields

open problems

open problems

- are standard alphabets closed under unio

open problems

- are standard alphabets closed under unio
- is every deterministic TM equivalent to a
  with tape alphabet equal to input alphabe

open problems

- are standard alphabets closed under unio
- is every deterministic TM equivalent to a with tape alphabet equal to input alphabe
- explore the complexity zoo

open problems

- are standard alphabets closed under unio
- is every deterministic TM equivalent to a with tape alphabet equal to input alphabe
- explore the complexity zoo
- translate and investigate other models

open problems

- are standard alphabets closed under unio
- is every deterministic TM equivalent to a with tape alphabet equal to input alphabe
- explore the complexity zoo
- translate and investigate other models

classical sets	sets with atoms (N, =)

open problems

- are standard alphabets closed under unio
- is every deterministic TM equivalent to a
  with tape alphabet equal to input alphabe
- explore the complexity zoo
- translate and investigate other models

classical sets	sets with atoms (N, =)
general Petri nets	elementary nets

open problems

- are standard alphabets closed under unio
- is every deterministic TM equivalent to a
  with tape alphabet equal to input alphabe
- explore the compl
- translate and inves

places:
atoms × (a finite set)

classical sets	sets with atoms (N, =)
general Petri nets	elementary nets

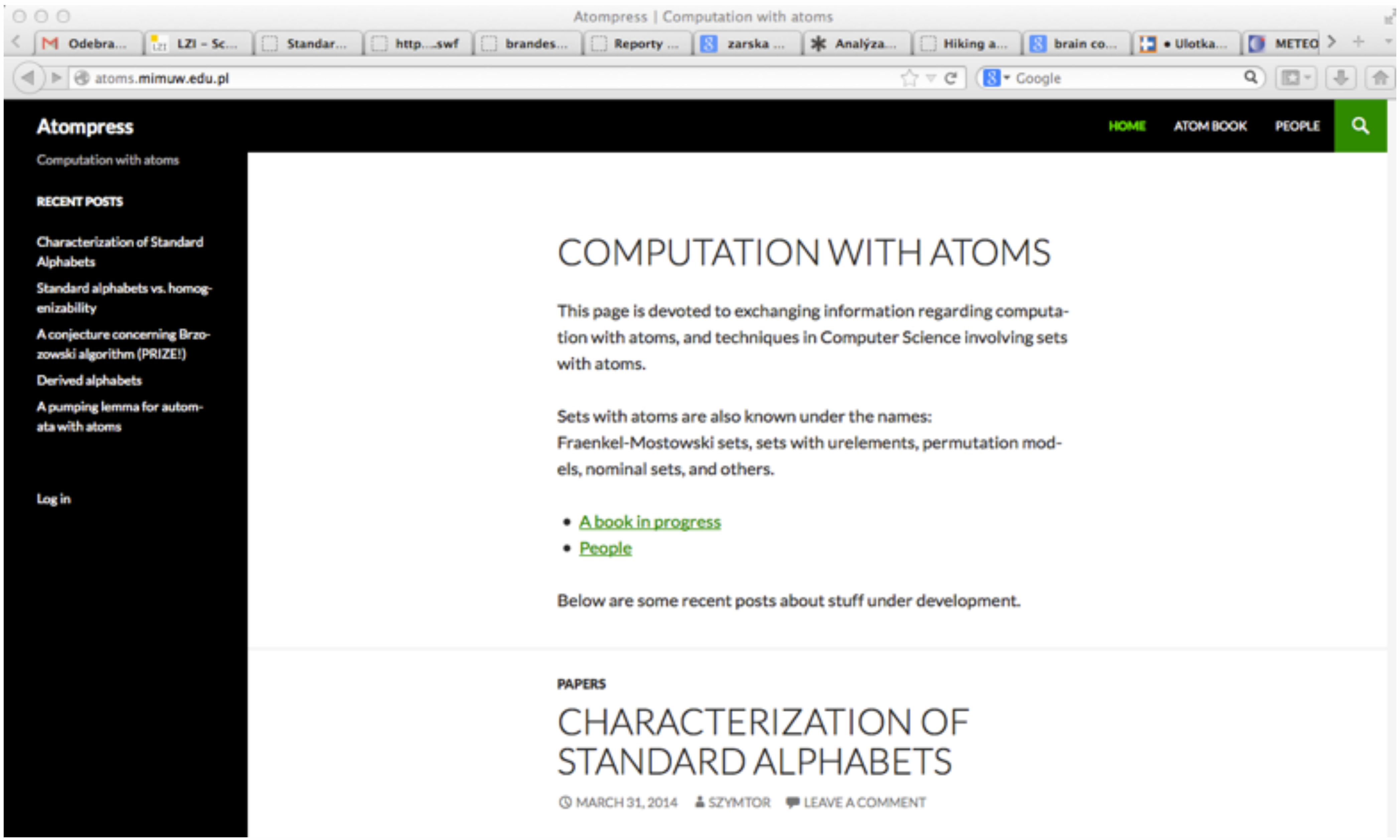
open problems

- are standard alphabets closed under unio
- is every deterministic TM equivalent to a
  with tape alphabet equal to input alphabe
- explore the compl
- translate and inves

classical sets	sets with atoms (N, =)
general Petri nets	elementary nets
data Petri nets	general Petri nets

places:
atoms × (a finite set)

visit our blog



thank you!