Ordering infinity: indexing and compressing regular languages

Nicola Prezza, Ca’ Foscari university of Venice, Italy

Joint work with: Nicola Cotumaccio (GSSI), Giovanna D’Agostino (uniud), Alberto Policriti (uniud), Jarno Alanko (university of Helsinki), Davide Martincigh (uniud)

Università
Ca’Foscari
Venezia

On the menu

1. **Foundations: a theory of ordered regular languages**

    a. Sorting NFAs.
    b. Wheeler languages.
    c. Sorting any regular language: partial co-lex orders
    d. Sortability hierarchies of regular languages

2. **Complexity**

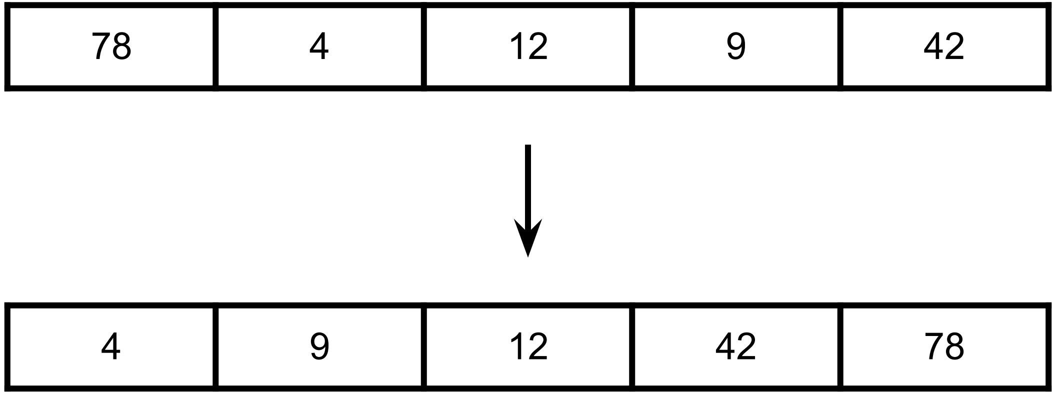
    a. Deciding the sortability of NFAs / regular languages
    b. Polynomial-time algorithms for sorting NFAs

3. **Open problems**

1.a Sorting Finite-state Automata

Sorting

Sorting is the algorithmic process of ordering the elements of a given set according to a specific order.

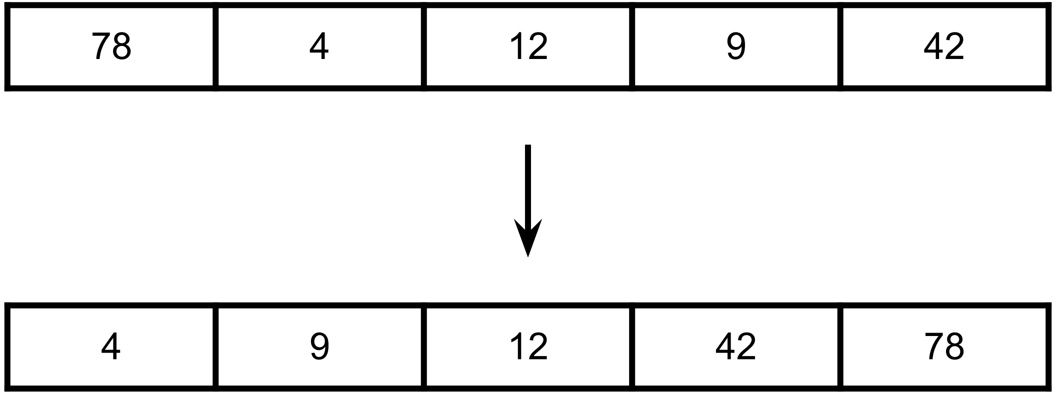


78	4	12	9	42

4	9	12	42	78

Sorting

Sorting is the algorithmic process of ordering the elements of a given set according to a specific order.



78	4	12	9	42

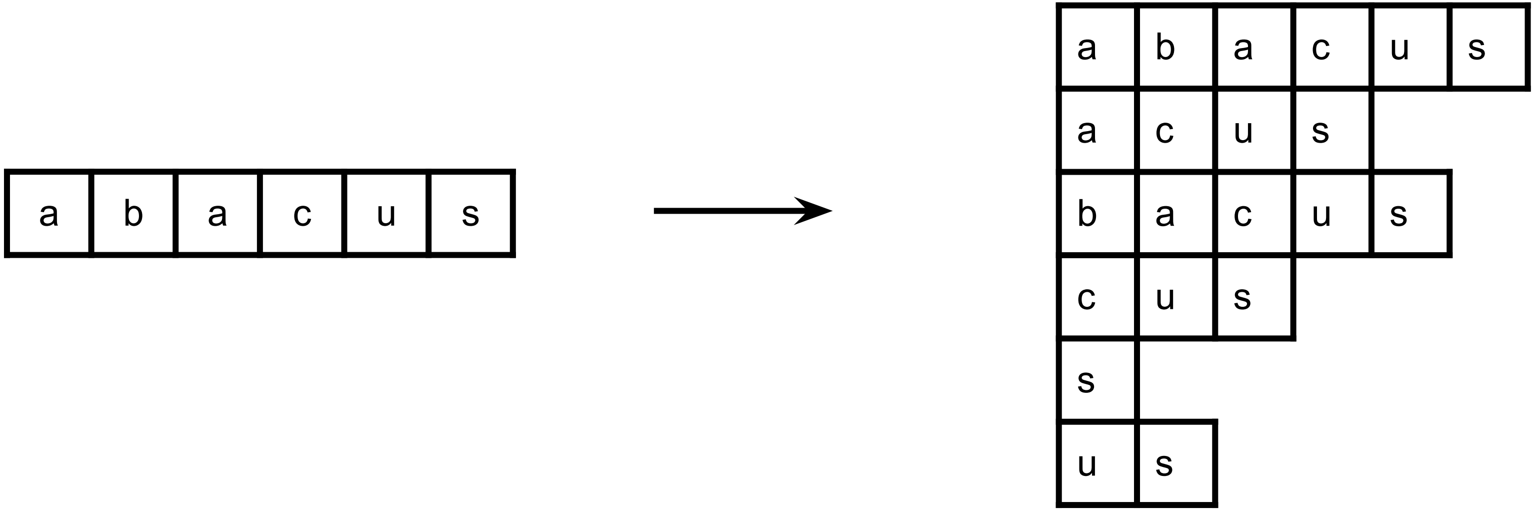
4	9	12	42	78

Example: integers, total order <. Benefits: the sorted list is

- Searchable (binary search; sorted list ≡ index)
- More compressible (delta-encoding: encode differences between consecutive integers)

Sorting

Not just integers. Other example: suffixes of a string

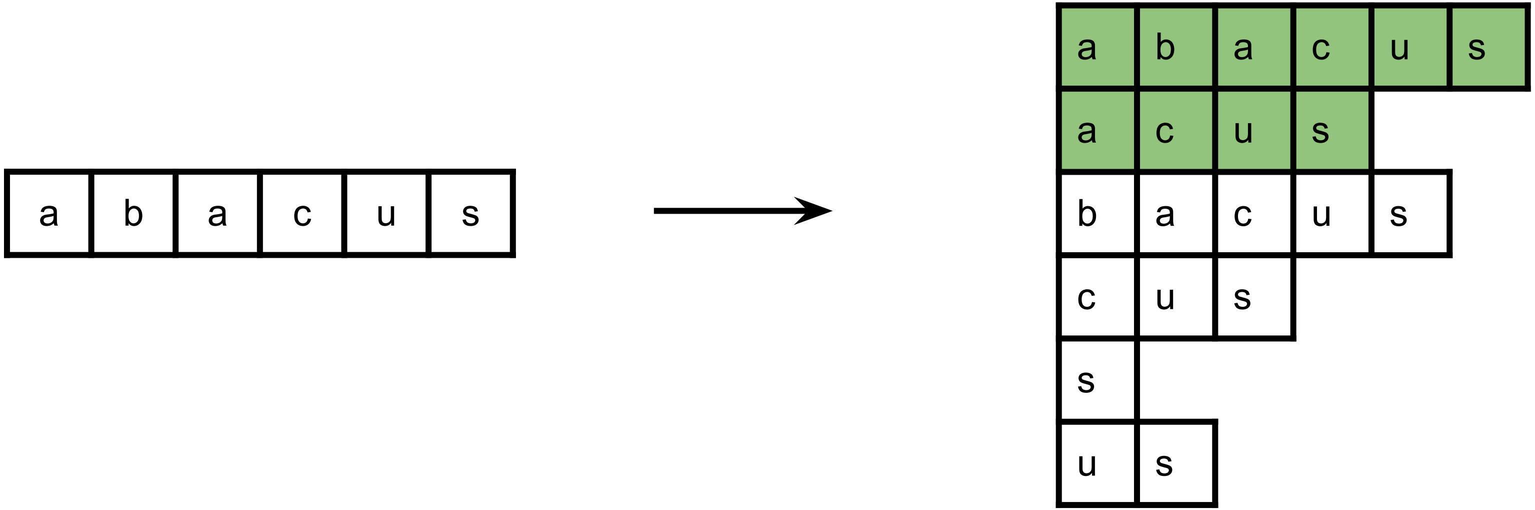


a	b	a	c	u	s

a	b	a	c	u	s
a	c	u	s		
b	a	c	u	s	
c	u	s			
s					
u	s				

Sorting

Not just integers. Other example: suffixes of a string



a	b	a	c	u	s

→

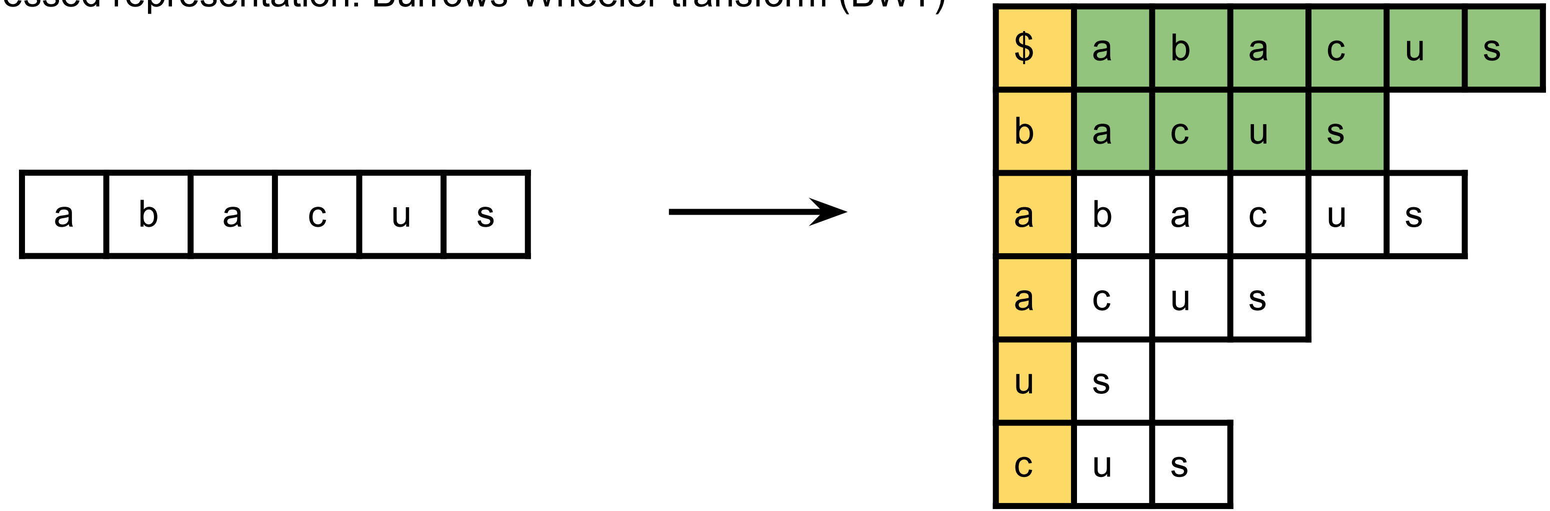
a	b	a	c	u	s
a	c	u	s		
b	a	c	u	s	
c	u	s			
s					
u	s				

Indexing and compression still hold!

- Indexing: suffixies prefixed by a word (e.g. “a”) form a range. Can be found, e.g. by binary search.

Sorting

Not just integers. Other example: suffixes of a string
compressed representation: Burrows-Wheeler transform (BWT)



a	b	a	c	u	s

$	a	b	a	c	u	s
b	a	c	u	s		
a	b	a	c	u	s	
a	c	u	s			
u	s					
c	u	s				

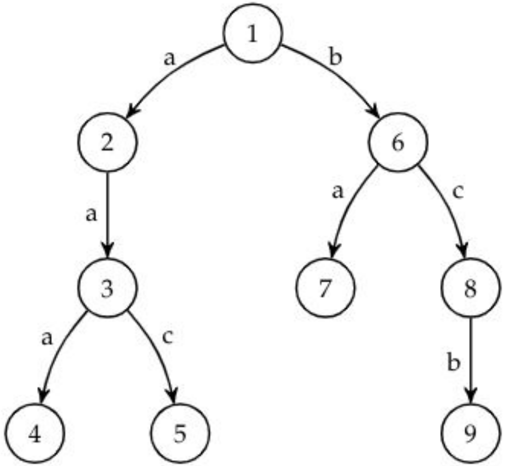
Indexing and compression still hold!

- Indexing: suffixies prefixed by a word (e.g. “a”) form a range. Can be found, e.g. by binary search.
- Compression: the index can be stored in compressed space (CSA [STOC’00], FM-index [FOCS’00]).

Sorting

Why stopping here?

- Finite sets of strings:
  - eBWT, [Mantaci et al. TCS’07]
  - Suffix tree of a labeled tree [Kosaraju, FOCS’89]
  - xBWT of a labeled tree [Ferragina et al., FOCS’05]



S =

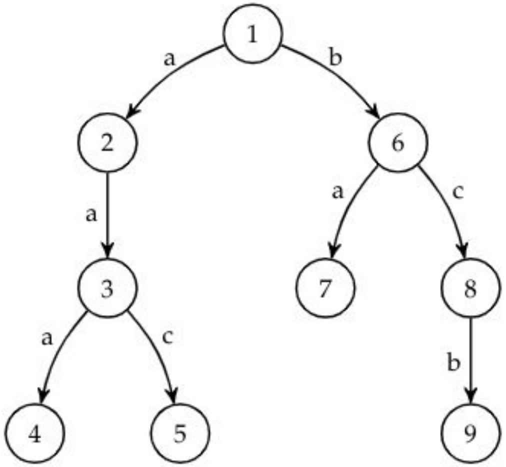
CATACCAT
ACCTCAT CCATAG
CTCATAC
TCATACCATA

Sorting

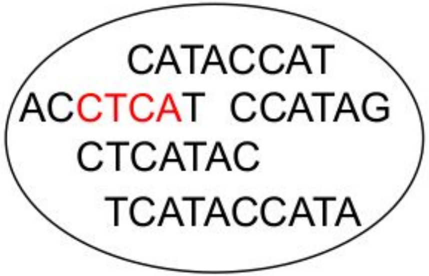
Why stopping here?

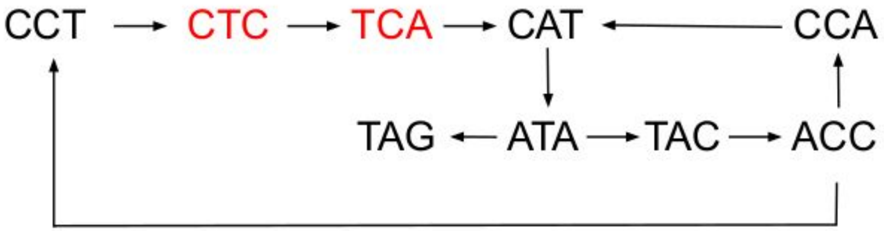
- Finite sets of strings:
  - eBWT, [Mantaci et al. TCS’07]
  - Suffix tree of a labeled tree [Kosaraju, FOCS’89]
  - xBWT of a labeled tree [Ferragina et al., FOCS’05]

- Infinite sets of strings:
  - BOSS: BWT of de Bruijn graphs [Bowe et al., WABI’12]
  - Wheeler graphs [Gagie et al. TCS’17]



S =


CATACCAT
ACCTCAT CCATAG
CTCATAC
TCATACCATA

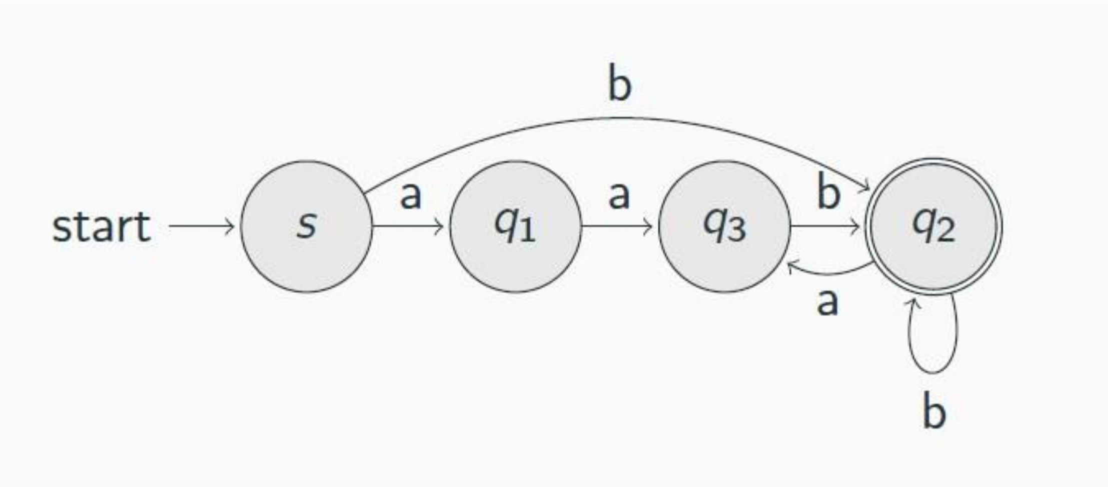


Wheeler graphs

[Gagie, Manzini, Sirén. "Wheeler graphs: A framework for BWT-based data structures." TCS’17]

WG = labeled graphs whose states can be sorted in a **total order** respecting the co-lex axioms:

1. in(u) < in(v) ⇒ u < v
2. u < v & (u,u’,a), (v,v’,a) ∈ E ⇒ u’ < v’



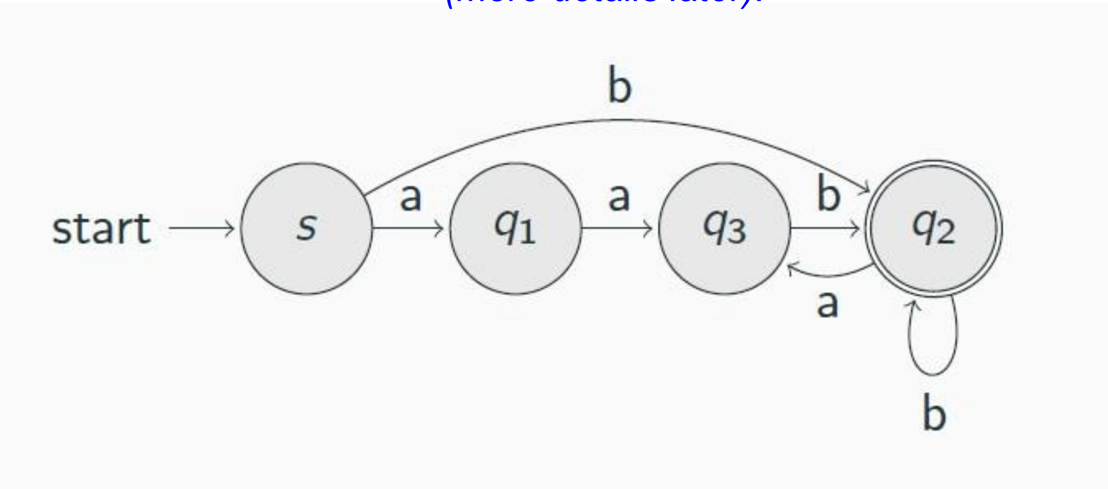
Wheeler graphs

[Gagie, Manzini, Sirén. "Wheeler graphs: A framework for BWT-based data structures." TCS’17]

WG = labeled graphs whose states can be sorted in a **total order** respecting the co-lex axioms:

1. in(u) < in(v) ⇒ u < v
2. u < v & (u,u’,a), (v,v’,a) ∈ E ⇒ u’ < v’

These two axioms are not the only way to define an indexable order over the NFA’s states (more details later).

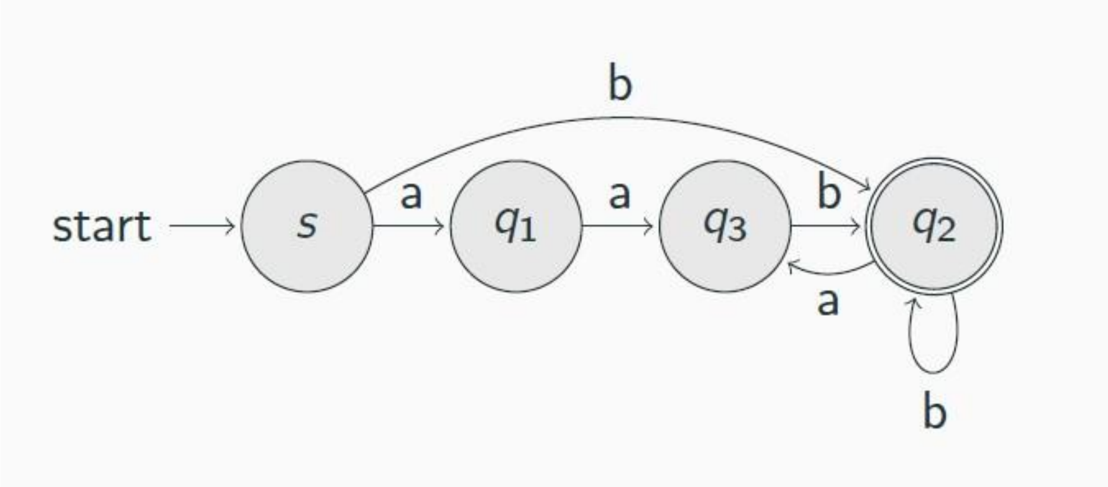


1.b From Sorting NFAs to Regular Languages

A new language-theoretical approach

New approach [Alanko, D’Agostino, Policriti, P. SODA’20]:

Let’s take a step back, and study the problem as **a problem on regular languages**.

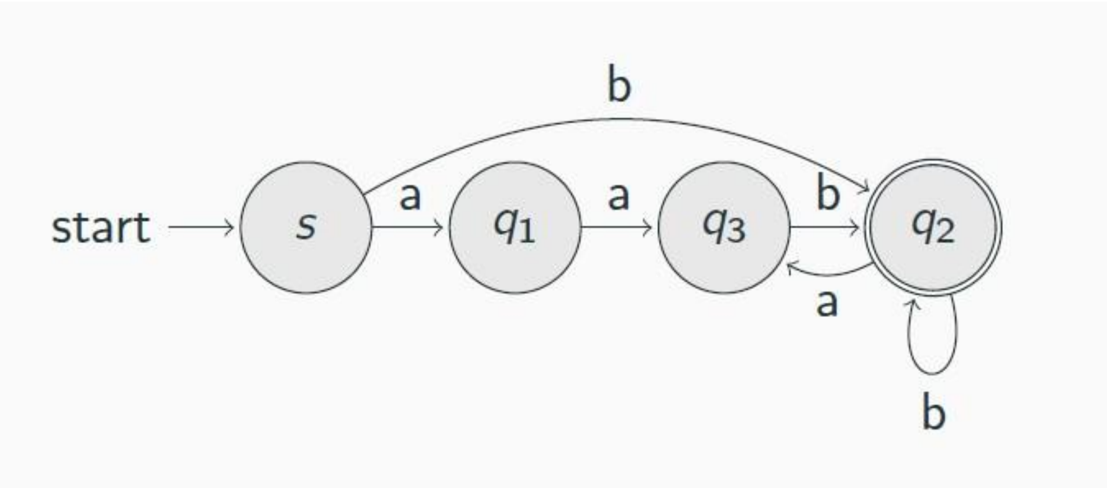


L = (ε|aa)b(ab|b)*

A new language-theoretical approach

New approach [Alanko, D’Agostino, Policriti, P. SODA’20]:

- L (regular, infinite) can be finitely represented as an NFA A.
- Sort co-lexicographically all prefixes of words in L.
- Map this information on A. What happens?



L = (ε|aa)b(ab|b)*

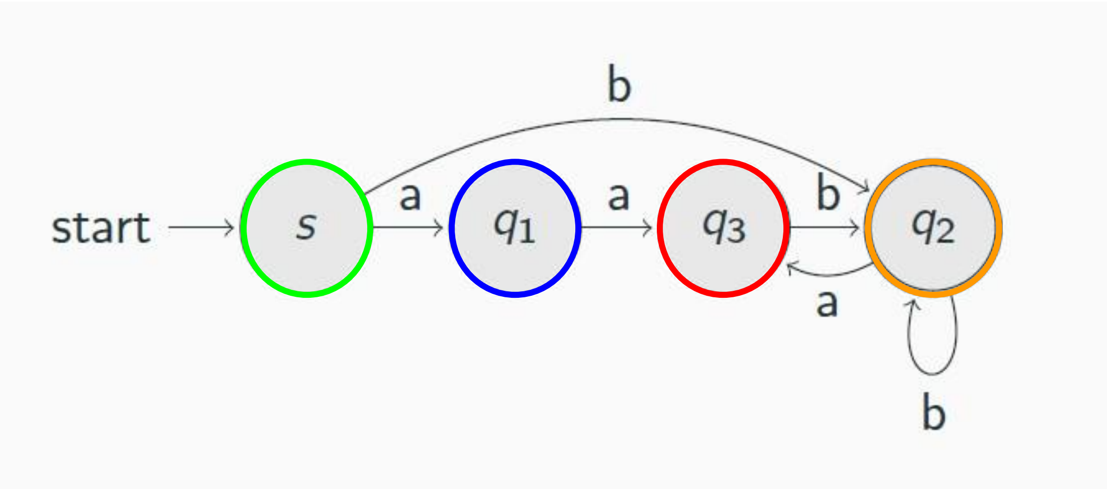
L =

ε
a
aa
ba
aaba
aababa
...
b
aab
bab
aabab
babab
…
bb
…
bbbb
....

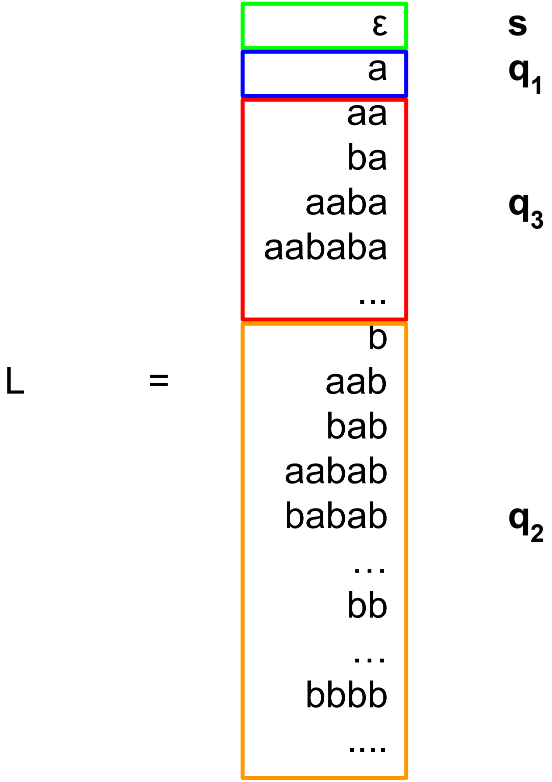
A new language-theoretical approach

New approach [Alanko, D’Agostino, Policriti, P. SODA’20]:

- L (regular, infinite) can be finitely represented as an NFA A.
- Sort co-lexicographically all prefixes of words in L.
- Map this information on A. What happens?



L = (ε|aa)b(ab|b)*


L =

ε — s

a — q_1

aa
ba
aaba — q_3
aababa
...

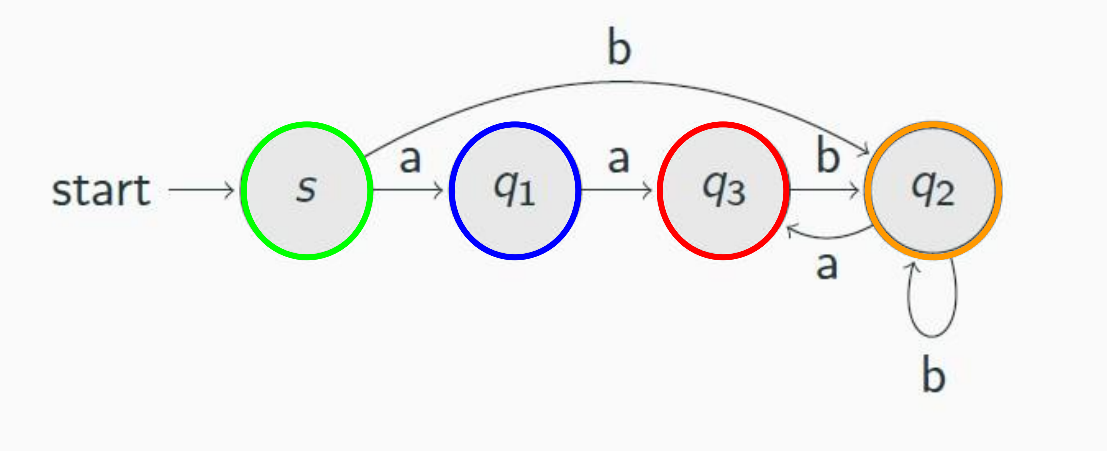
b
aab
bab
aabab
babab — q_2
…
bb
…
bbbb
....

A new language-theoretical approach

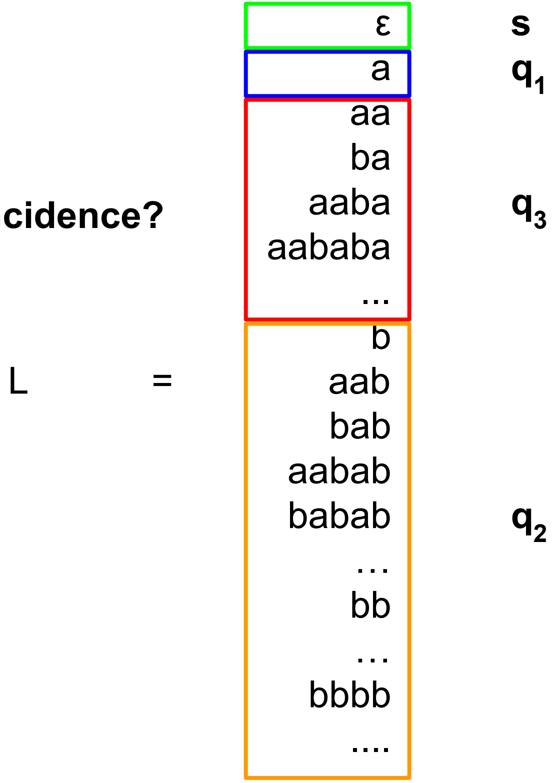
New approach [Alanko, D’Agostino, Policriti, P. SODA’20]:

- L (regular, infinite) can be finitely represented as an NFA A.
- Sort co-lexicographically all prefixes of words in L.
- Map this information on A. What happens?

States form intervals and we re-obtain the Wheeler order! coincidence?



L = (ε|aa)b(ab|b)*



L =

ε	**s**
a	**q_1**
aa ba aaba aababa …	**q_3**
b aab bab aabab babab … bb … bbbb ….	**q_2**

Wheeler languages

Not a coincidence. From [Alanko et al. SODA’20]:

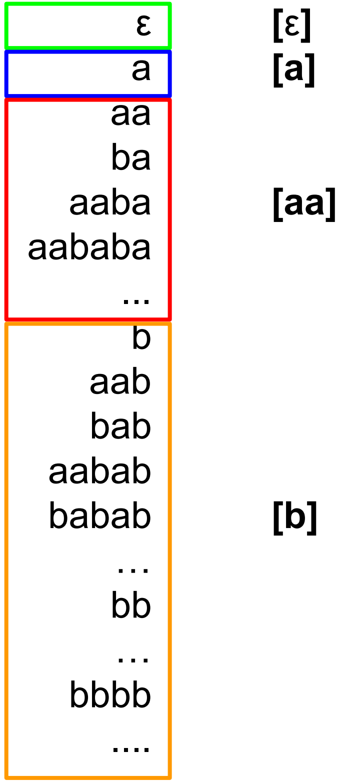
Theorem [Myhill-Nerode theorem for W. languages]:

A regular language is Wheeler

⇐⇒

*its Myhill-Nerode equivalence classes (≡ states of minimum DFA) form a **finite number of intervals in co-lex order.***

L = (ε|aa)b(ab|b)*



ε [ε]

a [a]

aa
ba
aaba [aa]
aababa
...

b
aab
bab
aabab
babab [b]
…
bb
…
bbbb
....

Wheeler languages

L = (ε|aa)b(ab|b)*

Not a coincidence. From [Alanko et al. SODA'20]:

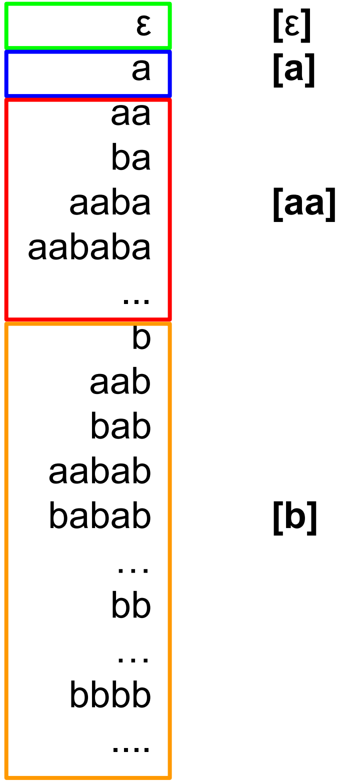
Theorem [Myhill-Nerode theorem for W. languages]:

A regular language is Wheeler

$\Longleftrightarrow$

its Myhill-Nerode equivalence classes (≡ states of minimum DFA) form a ***finite number of intervals in co-lex order.***

Wheeler languages = regular languages recognized by Wheeler NFAs
= regular languages recognized by Wheeler DFAs



ε [ε]
a [a]
aa
ba
aaba [aa]
aababa
...
b
aab
bab
aabab
babab [b]
…
bb
…
bbbb
....

Wheeler languages

Not a coincidence. From [Alanko et al. SODA'20]:

Theorem [Myhill-Nerode theorem for W. languages]:

A regular language is Wheeler

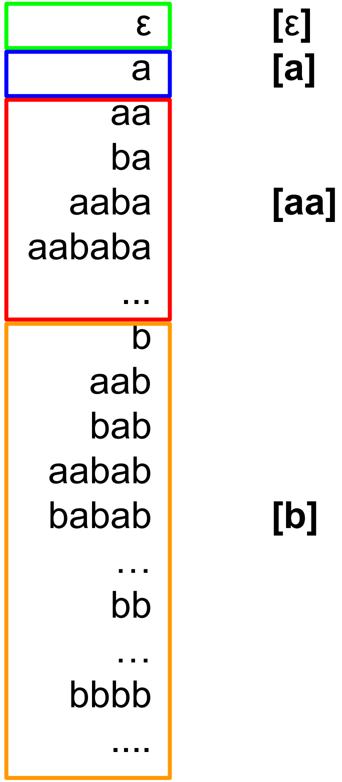
⇐⇒

its Myhill-Nerode equivalence classes (≡ states of minimum DFA) form a ***finite number of intervals in co-lex order.***

Wheeler languages = regular languages recognized by Wheeler NFAs
= regular languages recognized by Wheeler DFAs

More in detail: powerset determinization *always* turns a WNFA with n states into a WDFA with < 2n states.

L = (ε|aa)b(ab|b)*

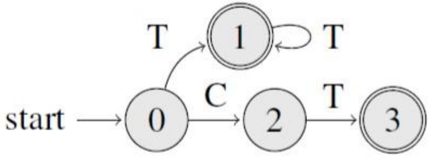


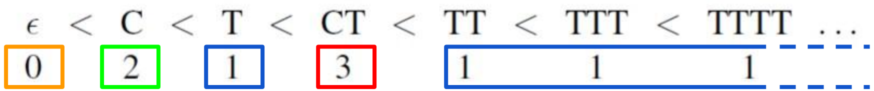
ε [ε]
a [a]
aa
ba
aaba [aa]
aababa
...
b
aab
bab
aabab
babab [b]
…
bb
…
bbbb
....

Wheeler languages

Note that also the following situation could occur:

- Some MN classes are split into pieces (in the example: class 1)
- Still, the number of MN *intervals* is *finite*





$$\epsilon < C < T < CT < TT < TTT < TTTT \dots$$

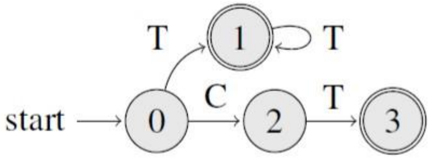
$$\boxed{0} \quad \boxed{2} \quad \boxed{1} \quad \boxed{3} \quad \boxed{1 \quad 1 \quad 1 \quad \dots}$$

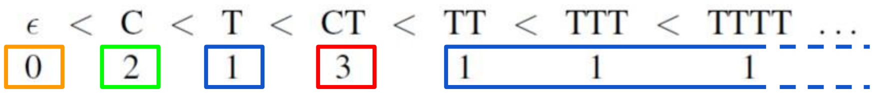
Finite number of MN intervals on the total order ≡ Wheeler language

Wheeler languages

Note that also the following situation could occur:

- Some MN classes are split into pieces (in the example: class 1)
- Still, the number of MN *intervals* is *finite*





Finite number of MN intervals on the total order ≡ Wheeler language

- In this case, the **DFA is not Wheeler**, but **the language is**.
- 5 intervals ≡ 5 states of a minimum *Wheeler DFA* for the language.
- Note: |min-DFA| < |min-WDFA| (the gap could be exponential)

Wheeler languages

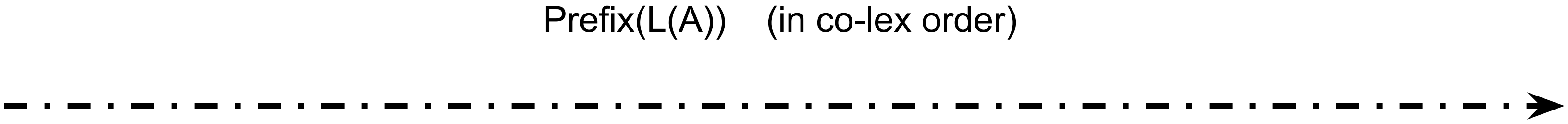
Another observation: previous examples concerned **DFAs**.

On **NFAs**, intervals could **overlap** in a prefix/suffix manner. In general, the picture becomes:

Wheeler languages

Another observation: previous examples concerned **DFAs**.

On **NFAs**, intervals could **overlap** in a prefix/suffix manner. In general, the picture becomes:

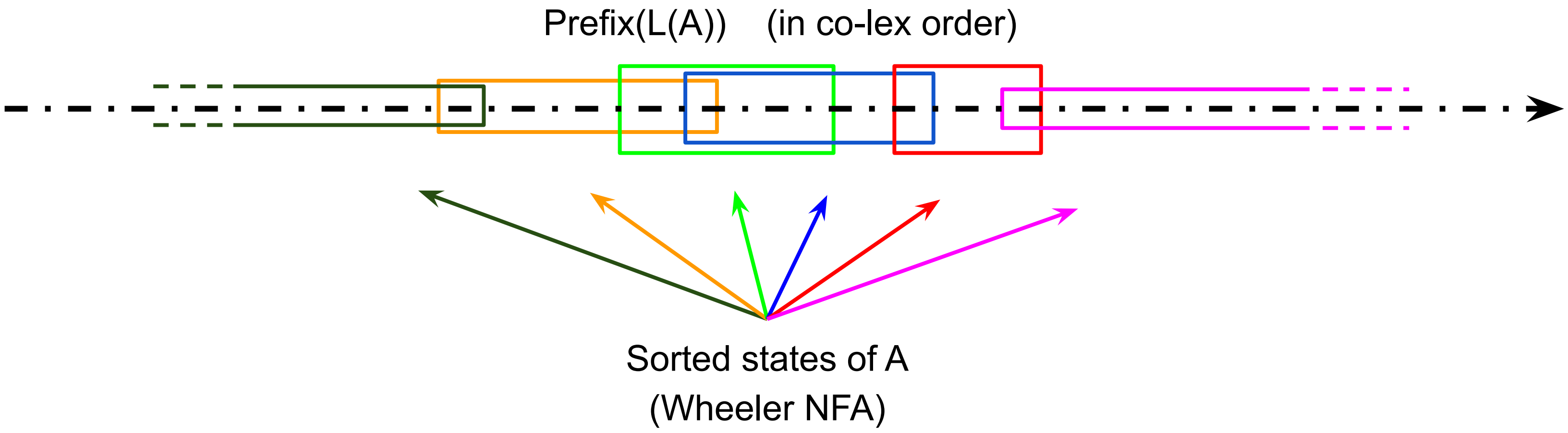


Prefix(L(A))   (in co-lex order)

Wheeler languages

Another observation: previous examples concerned **DFAs**.

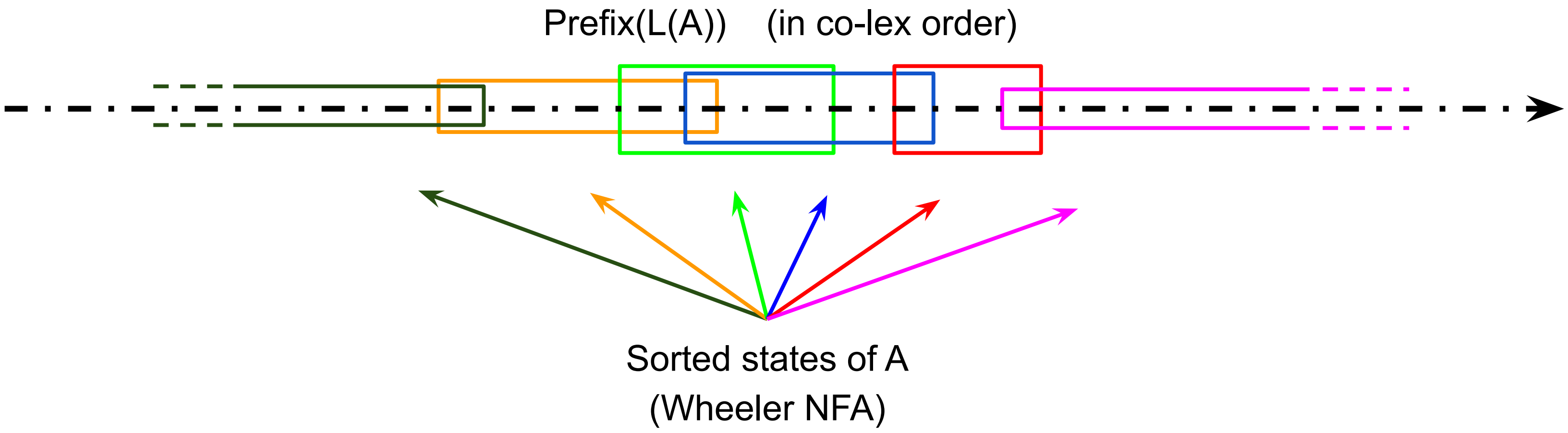
On **NFAs**, intervals could **overlap** in a prefix/suffix manner. In general, the picture becomes:



Wheeler languages

Another observation: previous examples concerned **DFAs**.

On **NFAs**, intervals could **overlap** in a prefix/suffix manner. In general, the picture becomes:



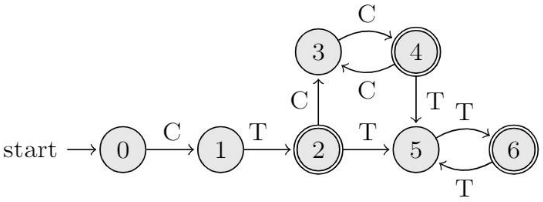
However, not all NFAs/languages are Wheeler! **can we index arbitrary NFAs/languages?**

1.c Partial co-lex orders

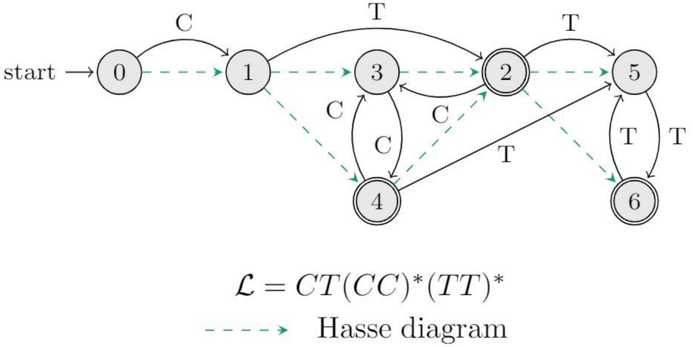
co-lex orders

Solution [Cotumaccio, P. SODA’21]: abandon total orders, embrace **partial orders**.

Result: any NFA admits a *partial co-lex order* of its nodes.



sort

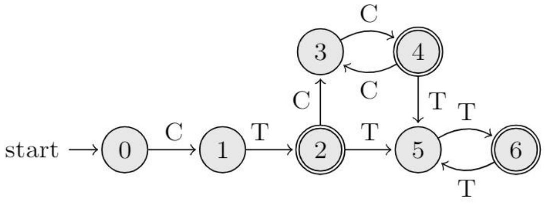


$\mathcal{L} = CT(CC)^*(TT)^*$

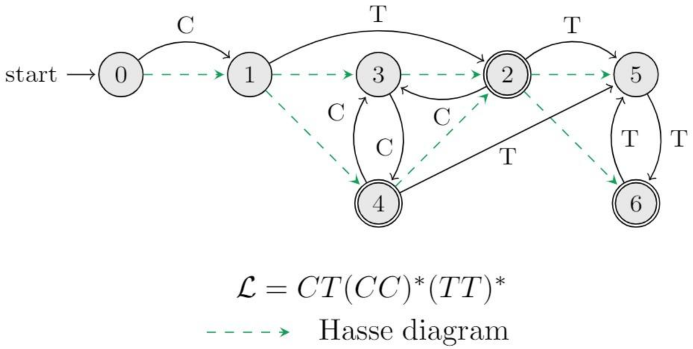
co-lex orders

Solution [Cotumaccio, P. SODA’21]: abandon total orders, embrace **partial orders**.

Result: any NFA admits a *partial co-lex order* of its nodes.



sort



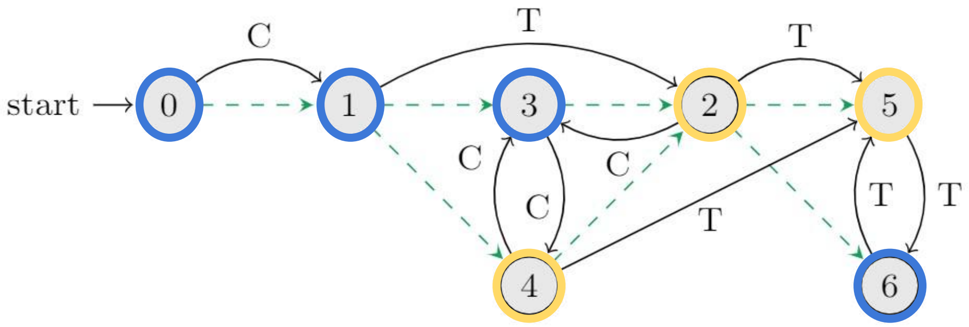
$\mathcal{L} = CT(CC)^*(TT)^*$

several < can be defined:

- **local** (axioms like in the Wheeler case, not necessarily unique),
- **global** (states = set of strings; extend co-lex order to sets of strings),
- **glocal** (reachability on the local definition, more details later)

co-lex orders

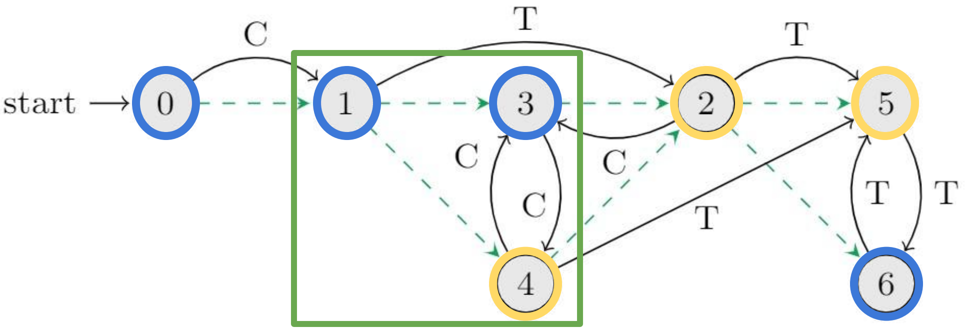
- We can partition states of A into **p** totally-ordered chains.
- The smallest **p = width(A)** is the order's **width** (in the example below, p = 2: {blue, yellow})



$$\mathcal{L} = CT(CC)^*(TT)^*$$

⇢ Hasse diagram

co-lex orders



$$\mathcal{L} = CT(CC)^*(TT)^*$$

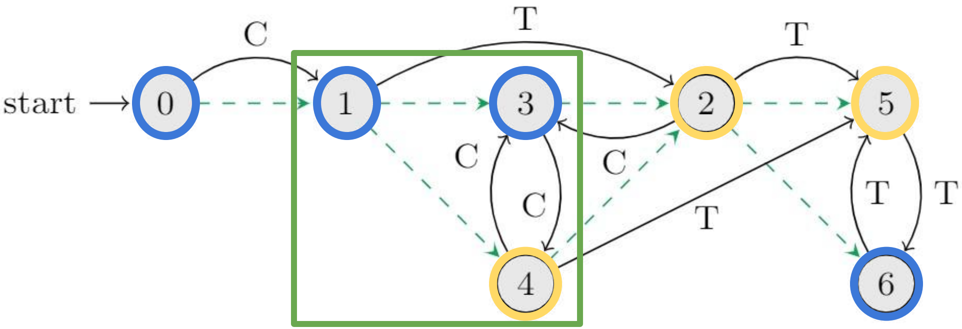
- - - - → Hasse diagram

Indexing and compression still work!

Indexing ≡ states reached by any string (“C”) always form a *convex set in the partial order*.

Convex set = p intervals on the p (totally-sorted) chains

co-lex orders



$$\mathcal{L} = CT(CC)^*(TT)^*$$

- - - - → Hasse diagram

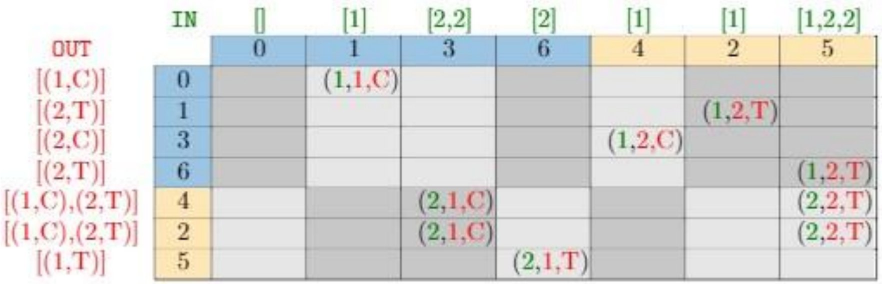
Indexing and compression still work!

Indexing ≡ states reached by any string (“C”) always form a *convex set in the partial order*.

Convex set = p intervals on the p (totally-sorted) chains

Compression: |BWT| = O(log p) bits per edge

BWT(A) = (IN,OUT)



	IN	[]	[1]	[2,2]	[2]	[1]	[1]	[1,2,2]
OUT		0	1	3	6	4	2	5
[(1,C)]	0		(1,1,C)					
[(2,T)]	1						(1,2,T)	
[(2,C)]	3					(1,2,C)		
[(2,T)]	6							(1,2,T)
[(1,C),(2,T)]	4			(2,1,C)				(2,2,T)
[(1,C),(2,T)]	2			(2,1,C)				(2,2,T)
[(1,T)]	5				(2,1,T)			

co-lex orders

Let n = number of states, m = number of edges.

[Cotumaccio, P. SODA'21] p = width(A) is a fundamental parameter for NFAs:

- Powerset determinization explodes with **2^p** (rather than 2^n)*

*consequence: NFA equivalence / universality (PSPACE-complete) are FPT w.r.t. p!

co-lex orders

Let n = number of states, m = number of edges.

[Cotumaccio, P. SODA'21] p = width(A) is a fundamental parameter for NFAs:

- Powerset determinization explodes with **2^p** (rather than 2^n)*
- NFA compression: O(**log p**) bits per edge (rather than log n)

*consequence: NFA equivalence / universality (PSPACE-complete) are FPT w.r.t. p!

co-lex orders

Let n = number of states, m = number of edges.

[Cotumaccio, P. SODA'21] p = width(A) is a fundamental parameter for NFAs:

- Powerset determinization explodes with **2^p** (rather than 2^n)*
- NFA compression: O(**log p**) bits per edge (rather than log n)
- NFA membership / pattern matching: O(**p^2**) time per character (rather than m)

*consequence: NFA equivalence / universality (PSPACE-complete) are FPT w.r.t. p!

1.d Sortability Hierarchies of Regular Languages

Width of a language

From [Cotumaccio, D’Agostino, Policriti, P. (submitted)]:

Definition Deterministic width $\text{width}^D(L)$ of L: smallest p such that there exists A DFA with:

- width(A) = p
- L(A) = L

Width of a language

From [Cotumaccio, D’Agostino, Policriti, P. (submitted)]:

Definition Deterministic width width^D(L) of L: smallest p such that there exists A DFA with:

- width(A) = p
- L(A) = L

Results:

- Non-unicity of the smallest-width DFA (Myhill-Nerode theorem for p-sortable languages)
- Characterization of a canonical smallest-width DFA: the *Hasse automaton* for L

Width of a language

From [Cotumaccio, D’Agostino, Policriti, P. (submitted)]:

Definition Nondeterministic width $\text{width}^N(L)$ of L. Smallest p such that there exists A NFA with:

- width(A) = p
- L(A) = L

Width of a language

From [Cotumaccio, D’Agostino, Policriti, P. (submitted)]:

Definition Nondeterministic width width^N(L) of L. Smallest p such that there exists A NFA with:

- width(A) = p
- L(A) = L

Definition The *width* of a regular language L is p = width(L) = width^N(L). We also say that L is p-sortable.

Width of a language

From [Cotumaccio, D’Agostino, Policriti, P. (submitted)]:

Definition Nondeterministic width width^N(L) of L. Smallest p such that there exists A NFA with:

- width(A) = p
- L(A) = L

Definition The *width* of a regular language L is p = width(L) = width^N(L). We also say that L is p-sortable.

Observation: width^N(L) = width^D(L) = 1 (total order) iff L is Wheeler.

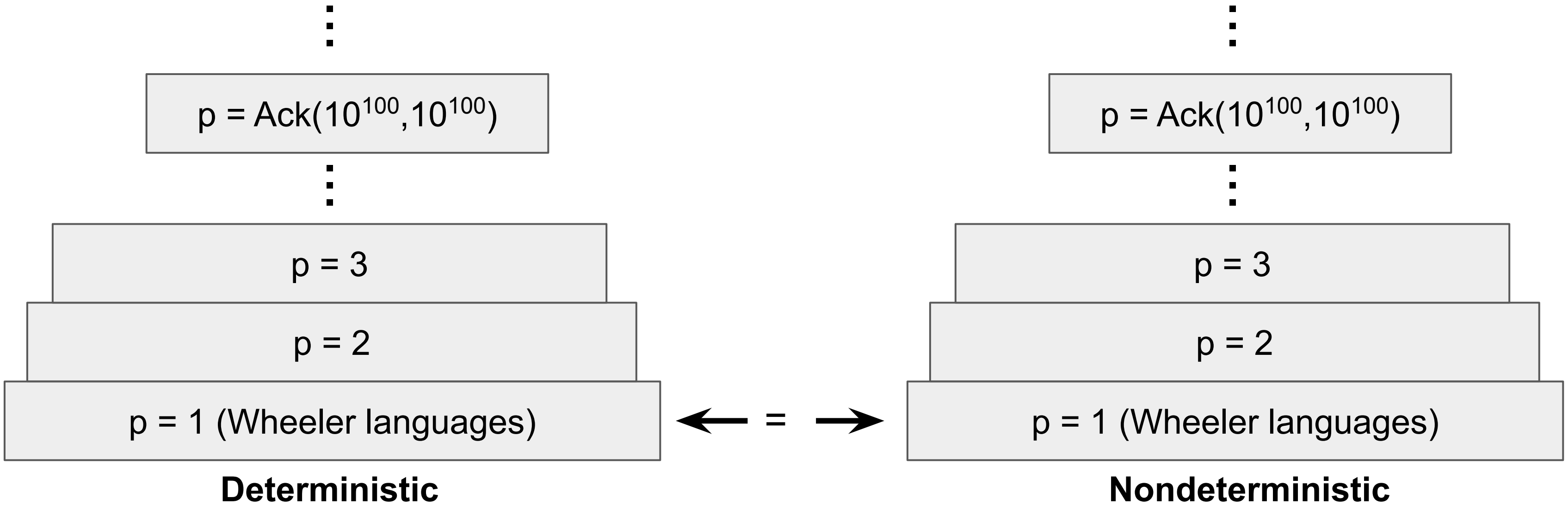
Width of a language

Which relations exist between width^N(L) and width^D(L)? We prove:

Width of a language

Which relations exist between width^N(L) and width^D(L)? We prove:

1. Both hierarchies are proper and do not collapse: for every p, there exists L such that width^N(L) = width^D(L) = p

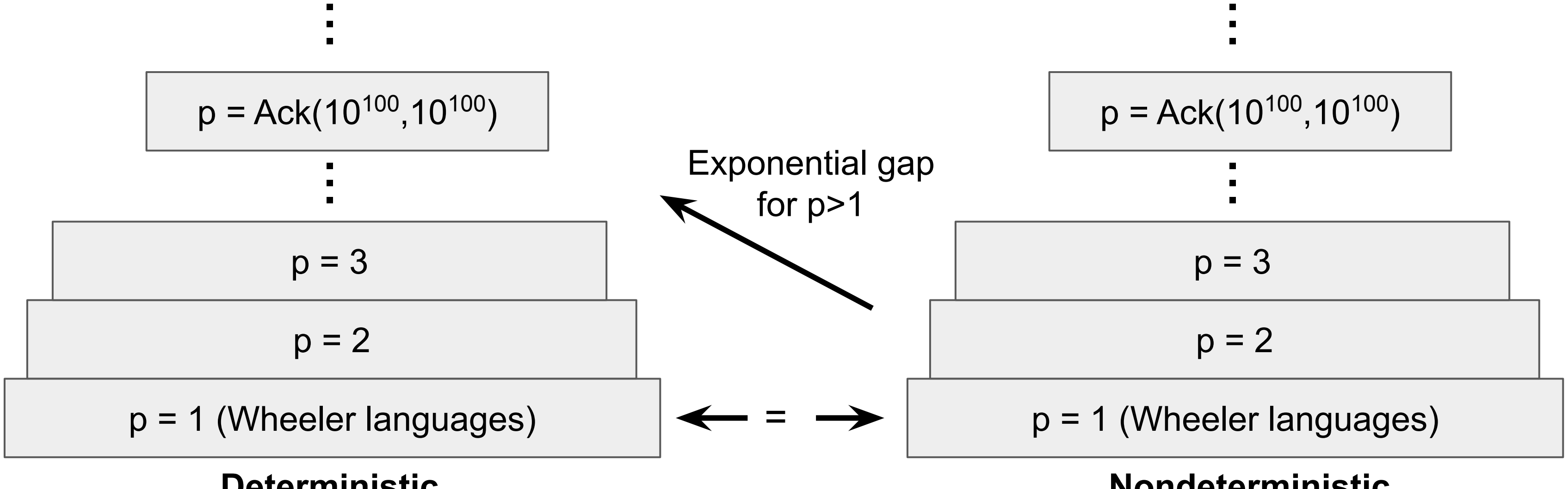


p = Ack(10^{100},10^{100})
p = 3
p = 2
p = 1 (Wheeler languages)
Deterministic
=
p = Ack(10^{100},10^{100})
p = 3
p = 2
p = 1 (Wheeler languages)
Nondeterministic

Width of a language

Which relations exist between width^N(L) and width^D(L)? We prove:

2. width^N(L) ≤ width^D(L) ≤ $2^{\text{width}^N(L)}$ - 1
3. There exist infinitely many L such that width^D(L) ≥ $e^{\sqrt{\text{width}^N(L)}}$


⋮
p = Ack(10^{100},10^{100})
⋮
p = 3
p = 2
p = 1 (Wheeler languages)
Deterministic
⋮
p = Ack(10^{100},10^{100})
⋮
p = 3
p = 2
p = 1 (Wheeler languages)
Nondeterministic
Exponential gap
for p>1
=

2.a Complexity Issues

Complexity issues

How hard is it to compute width(A) and width(L(A))?

Complexity issues

How hard is it to compute width(A) and width(L(A))?

First, a definition. Let q be a state of an NFA A.

Definition: I_q is the *language recognized by q*: set of strings labeling paths that connect the source of A to q.

Complexity issues

How hard is it to compute width(A) and width(L(A))?

First, a definition. Let q be a state of an NFA A.

Definition: *I_q* is the *language recognized by q*: set of strings labeling paths that connect the source of A to q.

Definition: an NFA A is *reduced* iff $q \neq q' \Rightarrow I_q \neq I_{q'}$

Complexity issues

How hard is it to compute width(A) and width(L(A))?

given / compute	**A: DFA**	**A: reduced NFA**	**A: NFA**
width(A)	$O(m^2 + n^{5/2})$ [1]	$O(n^6)$ [4]	NP-hard [2]*
width(L(A))	$n^{O(\text{width}(L(A))}$ [4]**	PSPACE-hard [3]*	PSPACE-hard [3]*

[1] Cotumaccio and P. On Indexing and Compressing Finite Automata. SODA'21.
[2] Gibney and Thankachan. On the hardness and inapproximability of recognizing Wheeler graphs. ESA'19
[3] D'Agostino, Martincigh, Policriti. Ordering regular languages: a danger zone. ICTCS'21
[4] Cotumaccio, D'Agostino, Policriti, P. Ongoing work.

* completeness holds in the Wheeler (p=1) case.
** note: in P for Wheeler L(A).

2.b Sorting / Indexing Algorithms

Sorting and Indexing

Recipe for indexing (optimally) an NFA: [Cotumaccio, P. 2021]:

1. Compute co-lex order < of smallest width.

2. Compute a smallest chain decomposition of (Q,<).
   $O(n^{5/2})$ time (reduction to maximum matching)

3. Build BWT of the NFA. O(m+n) time given the chain decomposition.

Sorting and Indexing

Recipe for indexing (optimally) an NFA: [Cotumaccio, P. 2021]:

1. Compute co-lex order < of smallest width.
2. Compute a smallest chain decomposition of (Q,<).
   $O(n^{5/2})$ time (reduction to maximum matching)
3. Build BWT of the NFA. $O(m+n)$ time given the chain decomposition.

Theorem [Cotumaccio, P. 2021]. (1) can be solved in $O(m^2)$ time on **DFAs**.

Theorem [Gibney, Thankachan. 2019]. (1) is NP-hard on **NFAs**!

Sorting and Indexing

Not all hope is lost, however. [Cotumaccio, D’Agostino, Policriti, P. Ongoing work]:

Definition (glocal order) Let $q \trianglelefteq q'$ iff $(q \leq_1 q_1 \leq_2 q_2 \ldots \leq_k q')$ for some co-lex pre-orders $\leq_1, \leq_2, \ldots, \leq_k$ and some states $q_1 \ldots q_{k-1}$.

Sorting and Indexing

Not all hope is lost, however. [Cotumaccio, D'Agostino, Policriti, P. Ongoing work]:

Definition (glocal order) Let q $\trianglelefteq$ q' iff (q $\leq_1$ q_1 $\leq_2$ q_2 … $\leq_k$ q') for some co-lex pre-orders $\leq_1$, $\leq_2$, …, $\leq_k$ and some states q_1 ... q_{k-1}.

Lemma On reduced NFAs, $\trianglelefteq$ is precisely the smallest-width co-lex pre-order ≤.

Sorting and Indexing

Not all hope is lost, however. [Cotumaccio, D'Agostino, Policriti, P. Ongoing work]:

Definition (glocal order) Let $q \trianglelefteq q'$ iff ($q \leq_1 q_1 \leq_2 q_2 \dots \leq_k q'$) for some co-lex pre-orders $\leq_1, \leq_2, \dots, \leq_k$ and some states $q_1 \dots q_{k-1}$.

Lemma On reduced NFAs, $\trianglelefteq$ is precisely the smallest-width co-lex pre-order $\leq$.
In general, *on any NFA*:

1. $\trianglelefteq$ is a partial (pre-)order
2. $\text{width}(\trianglelefteq) \leq \text{width}(\leq) = p$
3. $\trianglelefteq$ enables indexing
4. $\trianglelefteq$ can be computed in $O(n^6)$ time

Sorting and Indexing

Not all hope is lost, however. [Cotumaccio, D’Agostino, Policriti, P. Ongoing work]:

Definition (glocal order) Let $q \trianglelefteq q'$ iff ($q \leq_1 q_1 \leq_2 q_2 \ldots \leq_k q'$) for some co-lex pre-orders $\leq_1, \leq_2, \ldots, \leq_k$ and some states $q_1 \ldots q_{k-1}$.

Lemma On reduced NFAs, $\trianglelefteq$ is precisely the smallest-width co-lex pre-order $\leq$.
In general, *on any NFA*:

1. $\trianglelefteq$ is a partial (pre-)order
2. $\text{width}(\trianglelefteq) \leq \text{width}(\leq) = p$
3. $\trianglelefteq$ enables indexing
4. $\trianglelefteq$ can be computed in $O(n^6)$ time

We can index *any NFA* for the *optimal p* in polytime!

Sorting and Indexing

Not all hope is lost, however. [Cotumaccio, D’Agostino, Policriti, P. Ongoing work]:

Definition (glocal order) Let $q \trianglelefteq q'$ iff $(q \leq_1 q_1 \leq_2 q_2 \ldots \leq_k q')$ for some co-lex pre-orders $\leq_1, \leq_2, \ldots, \leq_k$ and some states $q_1 \ldots q_{k-1}$.

Lemma On reduced NFAs, $\trianglelefteq$ is precisely the smallest-width co-lex pre-order $\leq$.
In general, *on any NFA*:

1. $\trianglelefteq$ is a partial (pre-)order
2. width($\trianglelefteq$) $\leq$ width($\leq$) = p
3. $\trianglelefteq$ enables indexing
4. $\trianglelefteq$ can be computed in $O(n^6)$ time

We can index *any NFA* for the *optimal p* in polytime!

Note: we do not actually compute p, unless reduced NFA. Does not break NP-hardness of computing p (NFA used in the hardness proof is *not reduced*).

(infinite, unordered) list of open problems

1. Approximation algorithms for width(A) / width(L(A))
2. How does width(L) change with regexp operations?
3. Logical characterization of p-sortable languages (see Büchi's theorem: MSO ≡ REG)
4. Indexability lower bounds as a function of width(A) (fine-grained complexity)
5. Zoo of NFA orders (complexity, relations between different notions of width,...)
6. Algorithms for minimizing width(A) and/or number of states
7. Repetitive graph compression: run-length BWT / graph attractors
8. Dynamic data structures: maintain small width upon edge insertions/deletions
9. Generalizations: string-labeled edges, sorting context-free languages, ...
10. ...

Thank you! questions?