Space Efficient Merging of Compressed Indices

Lavinia Egidi and Giovanni Manzini
Wheeler graphs/automata

- Born as a unifying vision of BWT variants
  [Gagie, Manzini, Sirén, 2017]
  - BWT
  - XBWT
  - grafì di de Bruijn
  - ...

- Grown as a way to lift pattern matching from strings to languages
  [Alanko, D’Agostino, Policriti, Prezza, 2020]
  [Cotumaccio, Prezza, 2021]
Wheeler automaton

[Gagie, Manzini, Sirén, 2017]

• Let $A=(V,E,\Sigma,s,F)$ be an automaton with $L(A) \subseteq \Sigma^*$
  (totally ordered alphabet $\Sigma$)

• $A$ is Wheeler iff it admits a total order of $V$ s.t.
Wheeler automaton

[Gagie, Manzini, Sirén, 2017]

• Let $A=(V,E,\Sigma,s,F)$ be an automaton with $L(A) \subseteq \Sigma^*$ (totally ordered alphabet $\Sigma$)

• $A$ is Wheeler iff it admits a total order of $V$ s.t.
  - states with in-degree 0 are smallest

\[ s \xrightarrow{} w \]
Let $A=(V,E,\Sigma,s,F)$ be an automaton with $L(A) \subseteq \Sigma^*$ (totally ordered alphabet $\Sigma$)

$A$ is Wheeler iff it admits a total order of $V$ s.t.

- states with in-degree 0 are smallest
- for $(u,v,a),(u',v',a')$ transitions, if $a < a'$ then $v < v'$
Let $A=(V,E,\Sigma,s,F)$ be an automaton with $L(A) \subseteq \Sigma^*$ (totally ordered alphabet $\Sigma$)

$A$ is Wheeler iff it admits a total order of $V$ s.t.
- states with in-degree 0 are smallest
- for $(u,v,a),(u',v',a')$ transitions, if $a<a'$ then $v<v'$
- for $(u,v,a),(u',v',a)$ transitions, if $u<u'$ then $v \leq v'$
Searching for substrings of **ABRACADABRA**

We can use a **DFA**

Simple but **not** space efficient
Searching for substrings of ABRACADABRA

Compacted Trie

Suffix Tree

More space efficient!

“theoretically” space efficient
Searching for substrings of **ABRACADABRA**

We can use a **NFA**!
Every state initial & final

**Extremely** space efficient!
Searching for substrings of ABRACADABRA

Example:
Searching ABR
Searching for substrings of \textbf{ABRACADABRA}

Example:
Searching \textbf{ABR}
Searching for substrings of **ABRACADABRA**

Example:
Searching **ABR**
Searching for substrings of ABRACADABRA

Example:
Searching ABR
NFAs made simpler

“Naturally” assign a label to each state

A → B → R → A → C → A → D → A → B → R → A

_labels:
A → AB → ABR → ABRA → ABRAC → ABRACA → ABRACAD → ABRACADA → ABRACADAB → ABRACADABR → ABRACADABRA
NFAs made simpler

“Naturally” assign a label to each state

Arrange NFA states according to labels
Searching in a sorted NFA

Example:
Searching ABR
Searching in a sorted NFA

Example: Searching ABR
Searching in a sorted NFA

Example:
Searching ABR
Searching in a sorted NFA

Example:
Searching ABR

Two occurrences found!
NFAs made simpler

No need to store the state labels
NFAs made simpler

Add one arc for symmetry
A is Wheeler iff it admits a total order of $V$ s.t.
1) states with in-degree 0 are smallest
2) for $(u,v,a),(u',v',a')$ transitions, if $a < a'$ then $v < v'$
3) for $(u,v,a),(u',v',a)$ transitions, if $u < u'$ then $v \leq v'$
NFAs made simpler

A is Wheeler iff it admits a total order of \( V \) s.t.
1) states with in-degree 0 are smallest
2) for \((u,v,a),(u',v',a')\) transitions, if \(a<a'\) then \(v<v'\)
3) for \((u,v,a),(u',v',a)\) transitions, if \(u<u'\) then \(v \leq v'\)

Enough storing the edge labels:

\[\text{ABDBC}\$\text{RRAAAAA}\]

BWT of \((\text{ABRACADABRA})^R\)
NFAs made simpler

Each state has in-degree 1 and out-degree 1
NFAs made simpler

Each state has in-degree 1 and out-degree 1.
A colorful 8-state Wheeler automaton
Succinct representation of a Wheeler automaton
inspired by [Bowe, Onodera, Sadakane, Shibuya 2012]
Succinct representation of a Wheeler automaton

[Gagie, Manzini, Sirén, 2017]

• succinct representation
  \[2(|V|+|E|) + |E| \log|\Sigma| + |\Sigma| \log |E| + \text{l.o.t.}\]
• forward/backward traversal of an edge in time \(O(\log|\Sigma|)\)
Succinct representation of a Wheeler automaton

[Gagie, Manzini, Sirén, 2017]

- succinct representation
  \[2(|V|+|E|) + |E| \log |\Sigma| + |\Sigma| \log |E| + \text{l.o.t.}\]
- forward/backward traversal of an edge in time \(O(\log |\Sigma|)\)

Why does it work?
1) in a WA the set of states that are reached via a string is an interval (wrt the ordering)
Searching in a sorted NFA

Example:
Searching ABR
Succinct representation of a Wheeler automaton

[Gagie, Manzini, Sirén, 2017]

- succinct representation
  \[2(|V|+|E|) + |E| \log |\Sigma| + |\Sigma| \log |E| + \text{l.o.t.}\]
- forward/backward traversal of an edge in time \(O(\log |\Sigma|)\)

Why does it work?
1) in a WA the set of states that are reached via a string is an interval (with respect to the Wheeler order)
2) adding a character \(c \rightarrow \text{other interval}\)
Example: Searching ABR

Two occurrences found!
Succinct representation of a Wheeler automaton

[Gagie, Manzini, Sirén, 2017]

- succinct representation
  \[2(|V|+|E|) + |E| \log |\Sigma| + |\Sigma| \log |E| + \text{l.o.t.}\]
- forward/backward traversal of an edge in time \(O(\log |\Sigma|)\)

Why does it work?
1) in a WA the set of states that are reached via a string is an interval (with respect to the Wheeler order)
2) adding a character \(c \rightarrow\) other interval
3) in a total order an interval can be denoted by its endpoints
Recognizing Wheelerness of A

- is NFA A Wheeler? find an ordering:
  - decidable: NP-complete if $d \geq 5$
    [Gibney, Thankachan, 2019]
  - $\in P$ if $d \leq 2$
    [Alanko, D’Agostino, Policriti, Prezza, 2020]
L is wheeler if there is a Wheeler automaton A that decides it

- Some properties:
  - finite and cofinite languages are Wheeler
  - closed for:
    - intersections ✓
  - not closed for:
    - unions ✗
    - complements ✗
    - concatenations ✗
    - Kleene star ✗
Recognizing Wheelerness of $L$

[Alanko, D’Agostino, Policriti, Prezza, 2020]

- is $L$ Wheeler?
  - decidable: if it is expressed by an NFA
  - $\in$ P: if it is expressed by a DFA
  - minimum Wheeler DFA equivalent to a DFA with $n$ states has $\Omega(2^{n/4})$ states
Extension to arbitrary automata

[Cotumaccio, Prezza, 2021]

not Wheeler:
by 3), if $3 < 4$, $(3,4,a)$ and $(4,3,a) \Rightarrow 4 \leq 3$

A is Wheeler iff it admits a total order of $V$ s.t.
1) states with in-degree 0 are smallest
2) for $(u,v,a),(u',v',a')$ transitions, if $a< a'$ then $v < v'$
3) for $(u,v,a),(u',v',a)$ transitions, if $u < u'$ then $v \leq v'$
Extension to arbitrary automata

[ Cotumaccio, Prezza, 2021 ]

A is Wheeler iff it admits a total order of $V$ s.t.
1) states with in-degree 0 are smallest
2) for $(u,v,a),(u',v',a')$ transitions, if $a < a'$ then $v < v'$
3) for $(u,v,a),(u',v',a)$ transitions, if $u < u'$ then $v \leq v'$
Extension to arbitrary automata

\[ \text{p-sortable automaton} \iff \text{a co-lexicographic partial order of the states of width p=2} \]

A is Wheeler iff it admits a total order of \( V \) s.t.
1) states with in-degree 0 are smallest
2) for \( (u,v,a),(u',v',a') \) transitions, if \( a < a' \) then \( v < v' \)
3) for \( (u,v,a),(u',v',a) \) transitions, if \( u < u' \) then \( v \leq v' \)
Extension to arbitrary automata

$p$-sortable automaton $\iff$ chain decomposition $p=2$ chains

A is Wheeler iff it admits a total order of $V$ s.t.
1) states with in-degree 0 are smallest
2) for $(u,v,a),(u',v',a')$ transitions, if $a<a'$ then $v<v'$
3) for $(u,v,a),(u',v',a)$ transitions, if $u<u'$ then $v \leq v'$
Extension to arbitrary automata

Each chain is a total order and has properties 1)-3) of a Wheeler automaton

Why does it work?

1) in a WA the set of states that are reached via a string is an interval
2) adding a character $c \rightarrow$ other interval
3) total order $\rightarrow$ specify interval through its endpoints
Extension to arbitrary automata

Why does it work?
1) in a WA the set of states that are reached via a string is an interval
2) adding a character c → other interval
3) total order → specify interval through its endpoints

Each chain is a total order and has properties 1)-3) of a Wheeler automaton
essentially encode each chain as a WA

chain decomposition
p=2 chains

[Cotumaccio, Prezza, 2021]
Extension to arbitrary automata

[Cotumaccio, Prezza, 2021]

succinct representation
- $O(|E| \log |\Sigma| + \log p) + |V|$ space
- $O(p^2 \log(p|\Sigma|))$ time for forward traversal of edges

Wheeler automata [Gagie et al., 2017]
- succint representation, space:
  $2(|V|+|E|) + |E| \log|\Sigma|+ |\Sigma|\log |E| + \text{l.o.t.}$
- forward/backward traversal of edge in time $O(\log|\Sigma|)$
Extension to arbitrary automata

A $p$-sortable NFA with $|V|$ states via powerset construction

$\downarrow$

DFA $A'$ with $|V^*| \leq 2^p(|V|-p+1)-1$ states

[ Cotumaccio, Prezza, 2021 ]

Chain decomposition

$p=2$ chains
Merging deBruijn graphs and Wheeler automata

[ Egidi, Louza, Manzini, 2021 ]
accepted with minor revisions

Merging succinct representations:
given succinct representations of collections A and B,
obtain a succinct representation of $A \cup B$
Merging succinct representations:
given succinct representations of collections A and B,
obtain a succinct representation of \( A \cup B \)

expanding A and B to compute the union
Merging succinct representations:

given succinct representations of collections A and B,
obtain a succinct representation of $A \cup B$

Problems:
expanding A and B to compute the union
.require too much space
Merging deBruijn graphs and Wheeler automata

[Egidi, Louza, Manzini, 2021]
accepted with minor revisions

Merging succinct representations:
given succinct representations of collections A and B,
obtain a succinct representation of $A \cup B$

Problems:
expanding A and B to compute the union

😊 requires too much space

😢 if the representation is lossy, it’s not possible at all!
Merging deBruijn graphs and Wheeler automata

[EGIDI, LOUZA, MANZINI, 2021] accepted with minor revisions

Merging succinct representations:
given succinct representations of collections A and B, obtain a succinct representation of $A \cup B$

Do not expand! Merge directly! 😊
Merging deBruijn graphs and Wheeler automata

[Egidi, Louza, Manzini, 2021]
accepted with minor revisions

de Bruijn graph (k=3)

BOSS representation (lossy)

merging:
• $O(|E| k)$ time
• $4 |V|$ bits + $O(|\Sigma|)$ space

state of the art:
[Muggli et al, 2017,2019]
• same time
• space: $2(|E|\log|\Sigma|+|E|+|V|)+O(|\Sigma|)$
Merging deBruijn graphs and Wheeler automata

union of Wheeler automata

\[ A_0 \cup A_1 \]

[Egidi, Louza, Manzini, 2021]
accepted with minor revisions
union of Wheeler automata

$A_0$  
$s \xrightarrow{a} v \xrightarrow{a} \cdot$

$A_1$  
$s^* \xrightarrow{a} v^* \xrightarrow{a} w^*$

$A_0 \cup A_1$

Wheeler languages not closed for unions
Merging deBruijn graphs and Wheeler automata

[ Egidi, Louza, Manzini, 2021 ]
accepted with minor revisions

union of Wheeler automata

$A_0 \cup A_1$

union language Wheeler but union automaton not Wheeler:
$s < v, v^*$ and $v \neq v^*$

- $(s, v, a), (v^*, v^*, a) \Rightarrow v \leq v^*$
- $(s, v^*, a), (v, v, a) \Rightarrow v^* \leq v$

A is Wheeler iff it admits a total order of $V$ s.t.
1) states with in-degree 0 are smallest
2) for $(u, v, a), (u', v', a')$ transitions, if $a < a'$ then $v < v'$
3) for $(u, v, a), (u', v', a)$ transitions, if $u < u'$ then $v \leq v'$
Merging deBruijn graphs and Wheeler automata

- reduction to 2-SAT

\[ O(|E_0||E_1|) \]

- find a Wheeler order of \( A_0 \cup A_1 \)
- or report it doesn’t exist

(uses more space than the succinct representation)
Merging deBruijn graphs and Wheeler automata

[egidi, louza, manzini, 2021]
accepted with minor revisions

- refining algorithm: output a possibly smaller Wheeler automaton for the same language, or that no WO for the union automaton exists
Merging deBruijn graphs and Wheeler automata

[Egidi, Louza, Manzini, 2021]
accepted with minor revisions

- refining algorithm: output a possibly smaller Wheeler automaton for the same language, or that no WO for the union automaton exists

- time $O(|V|^2)$
- space $4|V|+o(|V|)$ bits
Conclusions

- Wheeler automata promising, also in view of recent work
- future work: extend our approach to generic automata through ideas in [Cotumaccio et al., 2021]