Regular Expression Matching

Inge Li Gørtz

Overview

- Regular expression matching
- Main techniques
- Parsing
- Lower bound
- Deterministic regular expressions
- Sparse regular expression matching
- Other parameterizations

Regular Expressions

- Regular expression over alphabet Σ:
  - $\alpha \in \Sigma \cup \{\epsilon\}$
  - If S and T are regular expressions then so are
    - $S \cdot T$
    - $S \,|\, T$
    - $S*$

- The language L(R) of a regular expression R:
  - $L(\alpha) = \{\alpha\}$
  - $L(S \cdot T) = L(S) \cdot L(T)$
  - $L(S \,|\, T) = L(S) \cup L(T)$
  - $L(S*) = \{\epsilon\} \cup L(S) \cup L(S)^2 \cup \ldots$

R = (a|ba)*

L(R) = {ε, a, aa, ba, aaa, aba, baa, aaaa, aaba, abaa, baaa, baba, …}

Regular Expression Matching

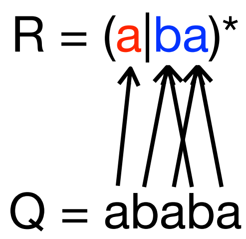
$R = (a|ba)^*$

$L(R) = \{\varepsilon, a, aa, ba, aaa, aba, baa, aaaa, aaba, abaa, baaa, baba, \ldots\}$

- Matching and parsing.

$R = (a|ba)^*$

$Q = ababa$



Matching:
Can Q be generated by R?

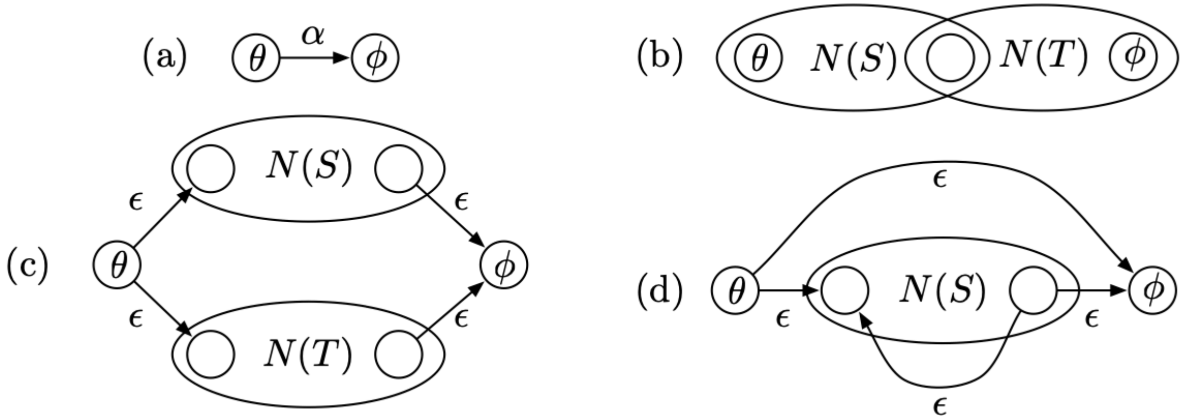
Parsing:
How can Q be generated by R?

- How fast can we solve regular expression matching/parsing for $|R| = m$ and $|Q| = n$?

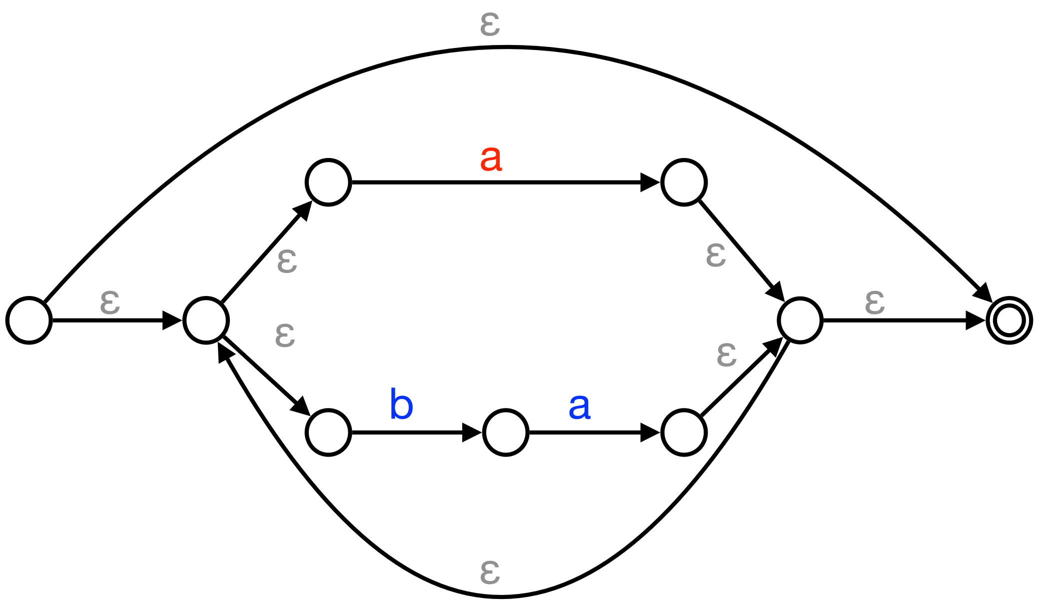
Applications

- Primitive in large scale data processing:
  - Internet Traffic Analysis
  - Protein searching
  - XML queries
- Standard utilities and tools
  - Grep and Sed
  - Perl

Thompson Automata



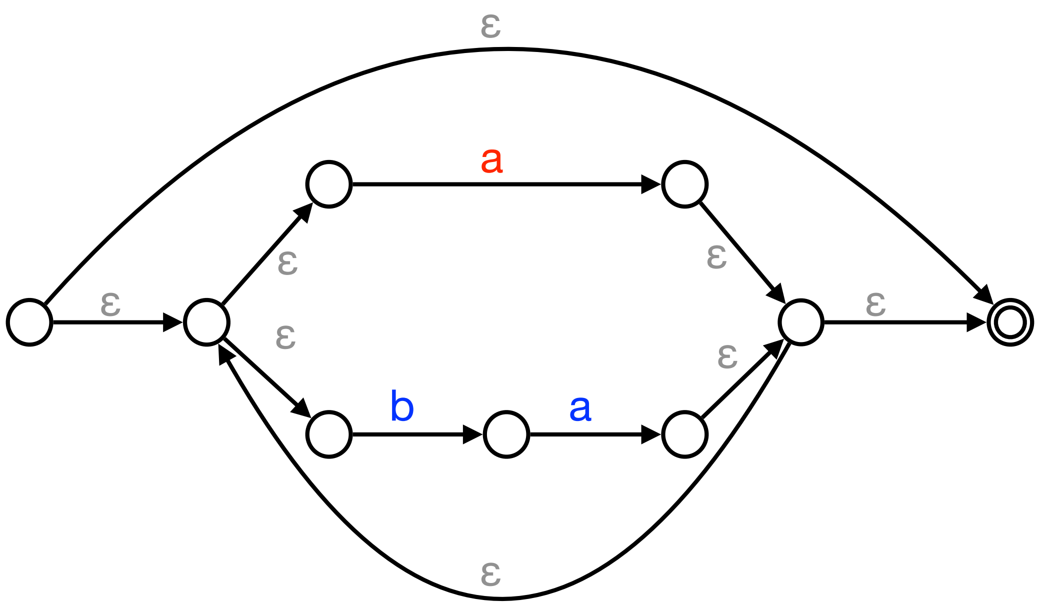
R = (a|ba)*



Regular Expression Matching

R = (a|ba)*

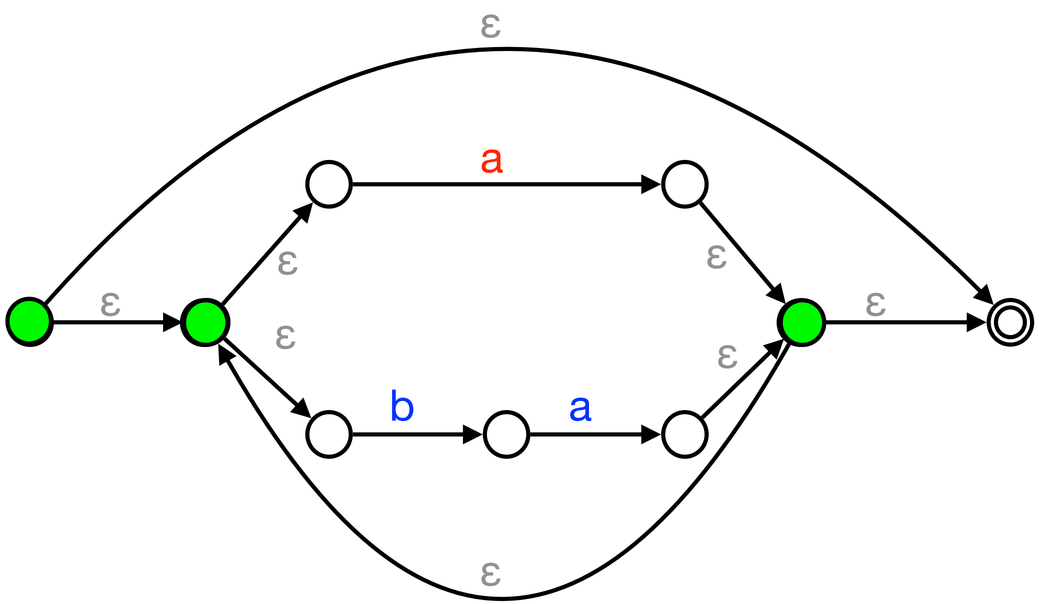
Q = ababa



Regular Expression Matching

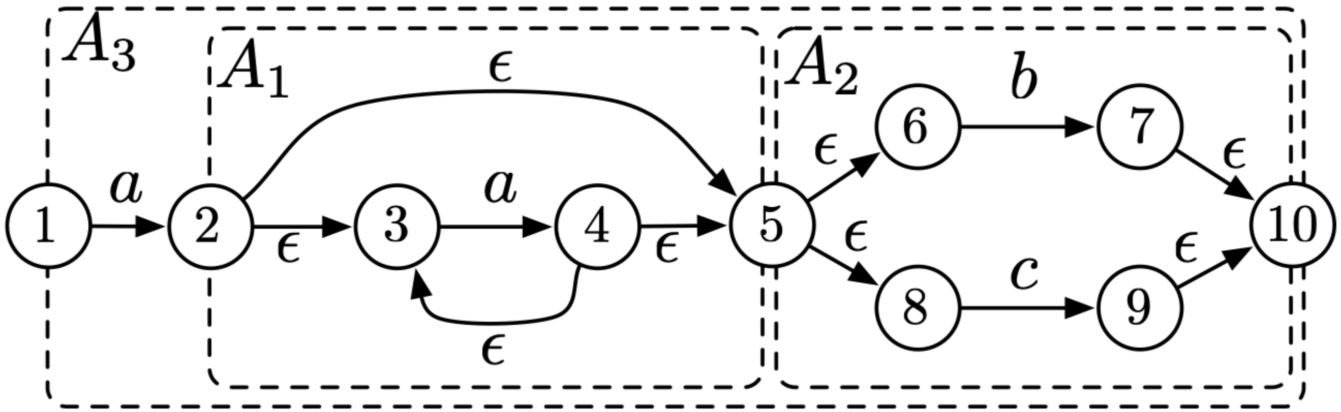
R = (a|ba)*

Q = ababa



- $O(nm)$ time and $O(m)$ space. [Thompson 1968]
- $O(nm/\log n)$ time and space. [Myers 1992]
- $O(nm/\log n)$ and linear space. [Bille 2006]
- Polylogarithmic improvements and linear space. [Bille and Farach-Colton 2008], [Bille and Thorup 2009]
- Lower bound assuming SETH [Backurs and Indyk 2016]

NFA Decomposition

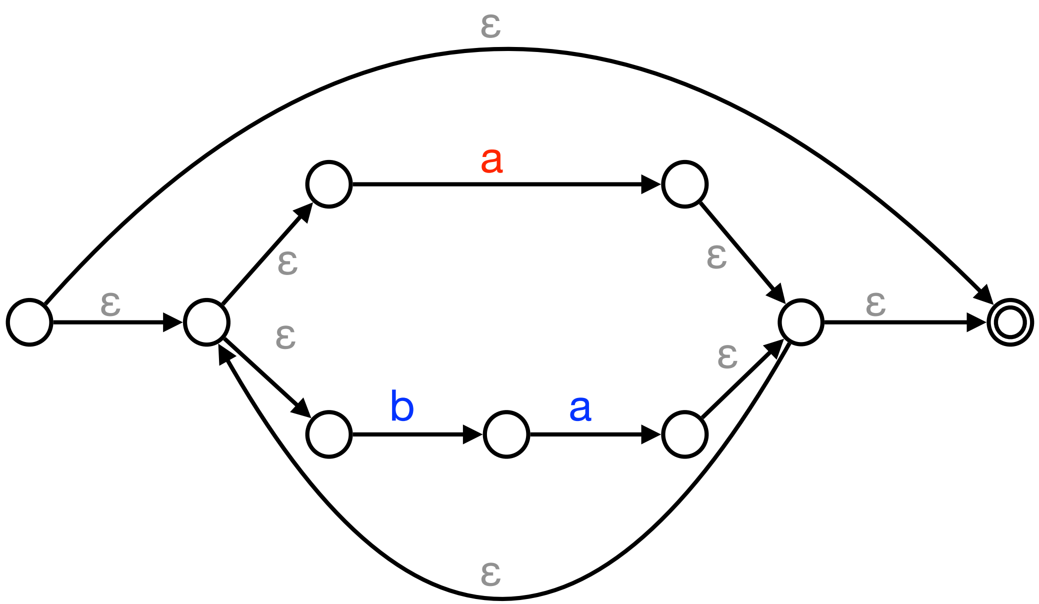


- Decompose NFA into tree of O(m/x) micro TNFAs with at most x states. Each micro TNFA overlaps with enclosing micro TNFA in 2 states.
- To do state-set transition using state-set simulation for micro TNFAs process micro TNFAs in topological order twice. Propagate reachable overlapping states.
- Use tabulation or word-level parallelism to perform micro TNFA simulation.

Regular Expression *Parsing*

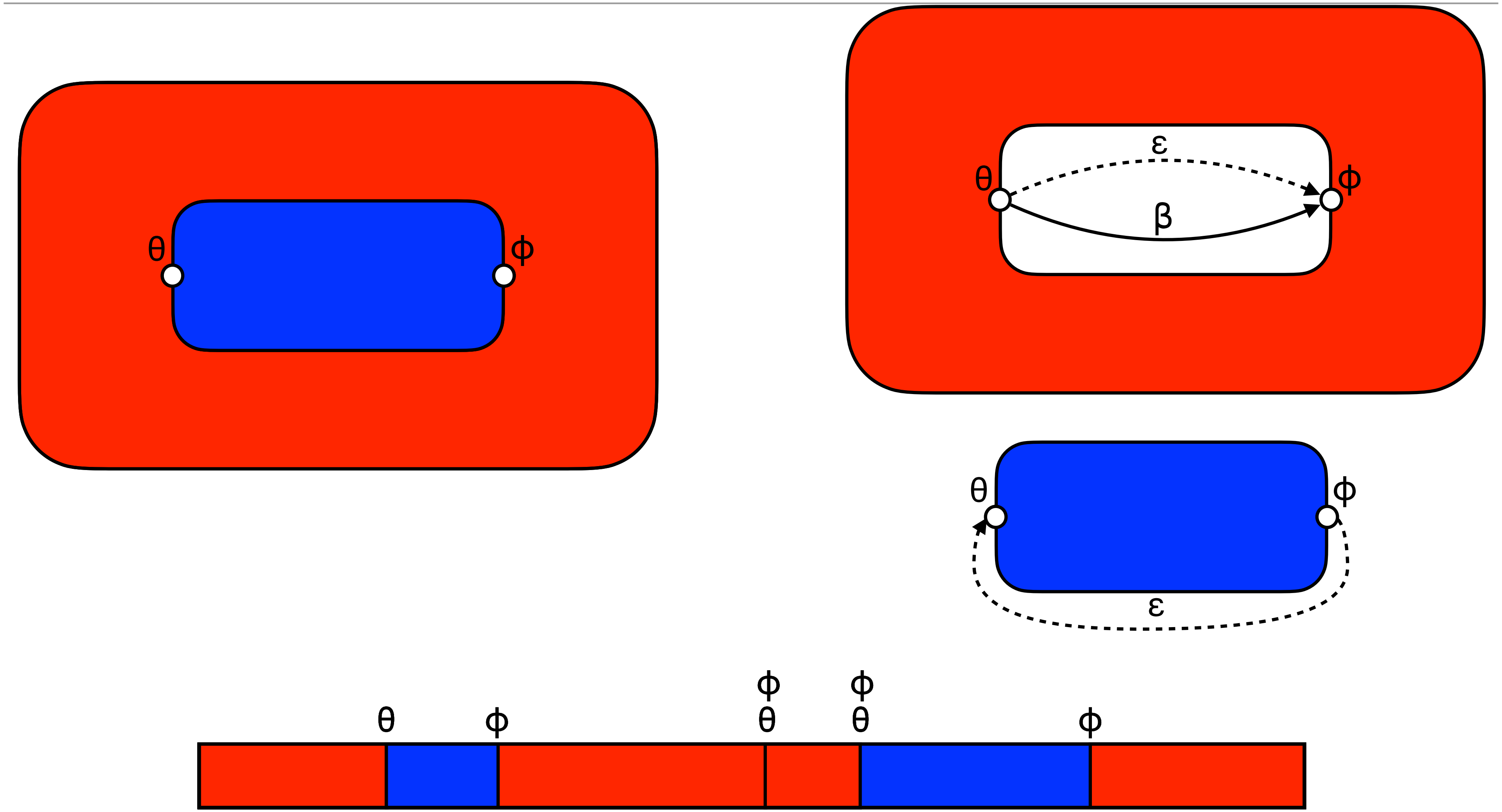
R = (a|ba)*

Q = ababa



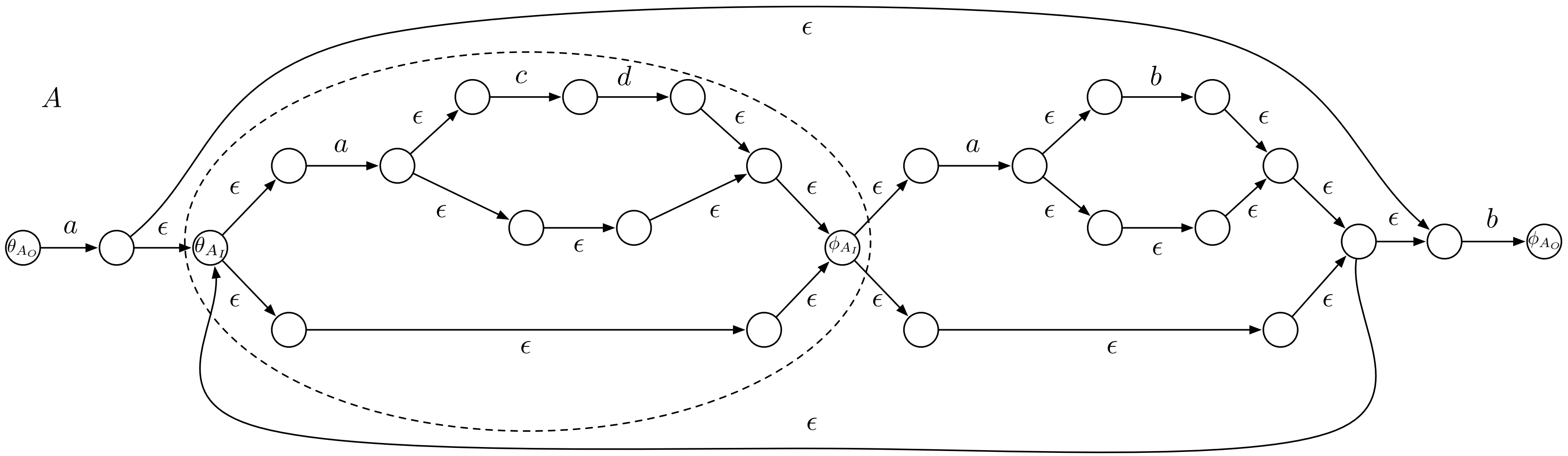
- $O(nm)$ space (backtracking).
- [Bille and G. 2019] $O(n+m)$ space

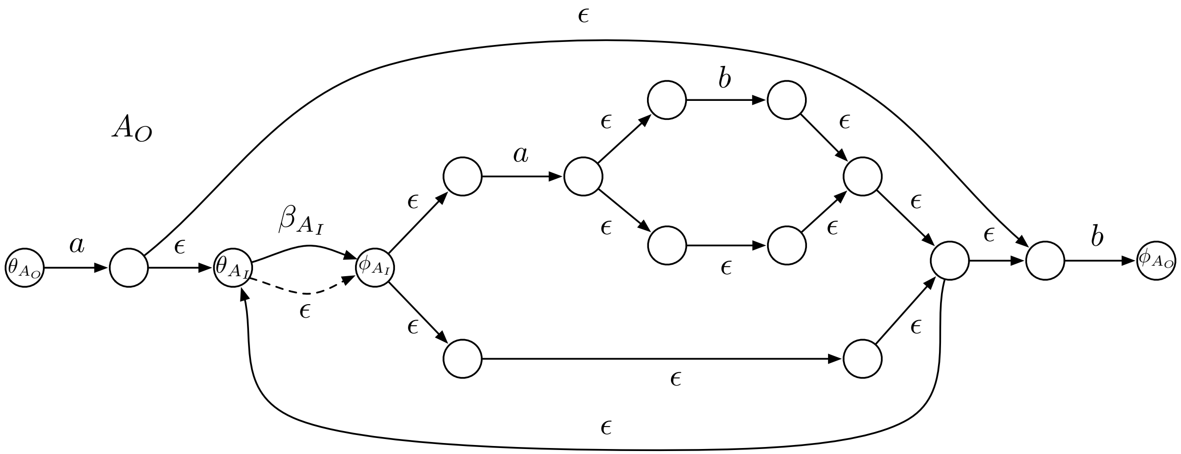
Divide

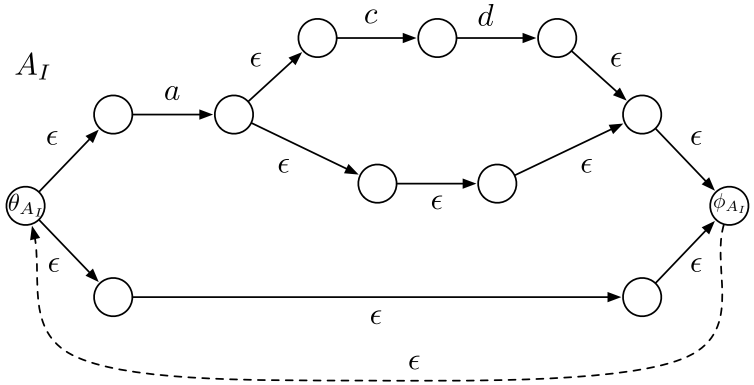


- O(nm) time and O(n+m) space.

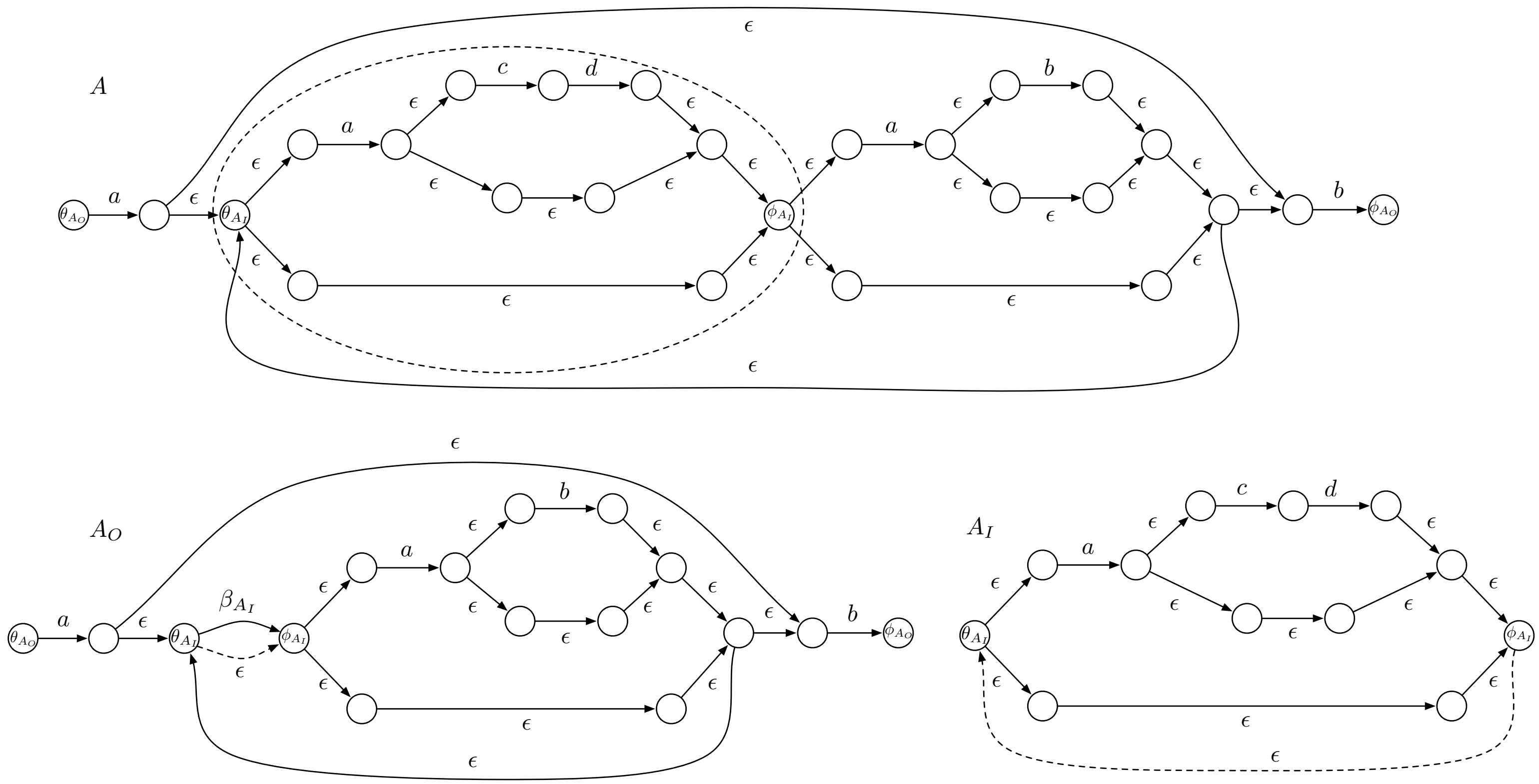
Regular Expression Parsing

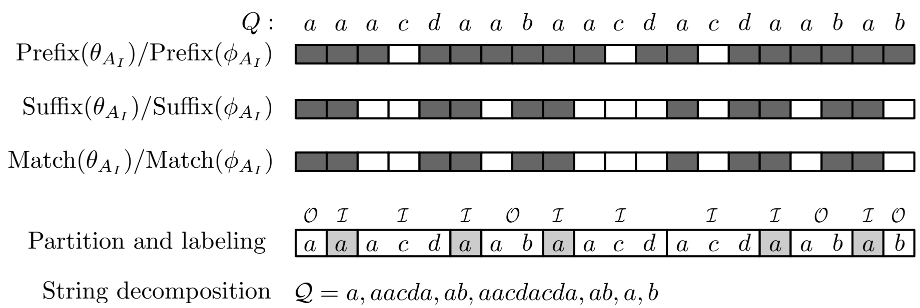




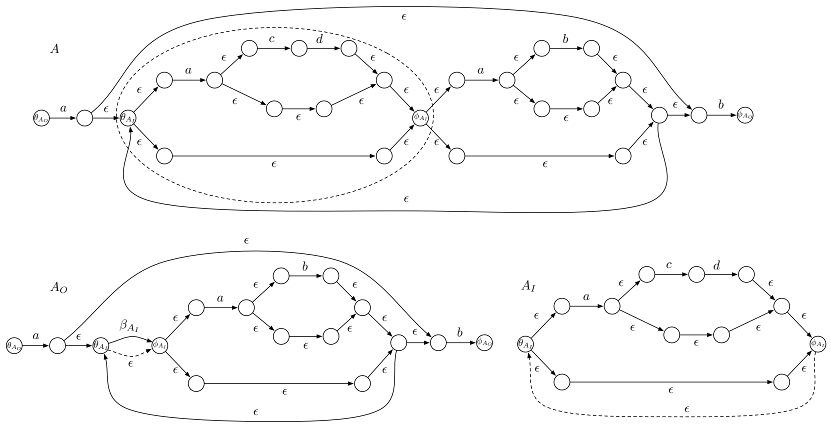


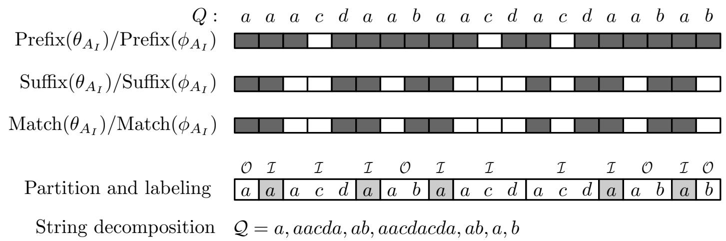
Regular Expression Parsing





Regular Expression Parsing





- General technique to convert regular expression matching algorithms to solve regular expression parsing at no asymptotic cost.
  - For *almost all* existing faster regular expression matching: same time and *linear* space for regular expression parsing.

Lower bound

- Orthogonal vectors problem (OVP). Given $A, B \subseteq \{0,1\}^d$, where $|A| = M$ and $|B| = N$, determine if there exists $a \in A$ and $b \in B$ such that $a \cdot b = 0$.

- Example.
  - $A = \{0001, 0101, 1000\}$, $B = \{0111, 1001\}$
  - Answer is YES: $1000 \cdot 0111 = 0$

- Conditional lower bound [Williams 2005, Bringmann and Künnemann 2015]. Assuming SETH there exists no $O((MN)^{1-\epsilon})$ time algorithm for $\epsilon > 0$ for OVP. Here $M = \Theta(N^\alpha)$ for $\alpha \in (0,1]$ and $d = \omega(\log N)$.

- Backurs and Indyk [2016]. Assuming SETH there exists no $O((nm)^{1-\epsilon})$ time algorithm for $\epsilon > 0$ for regular expression matching.

Lower bound

- Regular expression ***pattern*** matching reduction from OVP.
  - Construct regular expression P and string Q s.t.

    a substring of Q can be derived from $P \Leftrightarrow$ the answer to OVP is yes.
  - $\Sigma = \{x, y\}$
  - Time $O(Nd)$ and $|P| = \Theta(Md), |Q| = \Theta(Nd)$.

- Regular expression matching reduction.

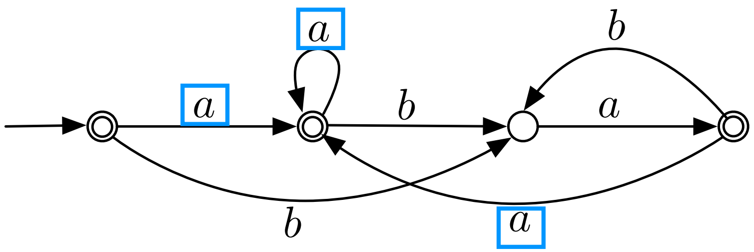
$$R = \left(\bigcup_{j=1}^{|Q|}(x^*y^*)\right) \cdot P \cdot \left(\bigcup_{j=1}^{|Q|}(x^*y^*)\right)$$

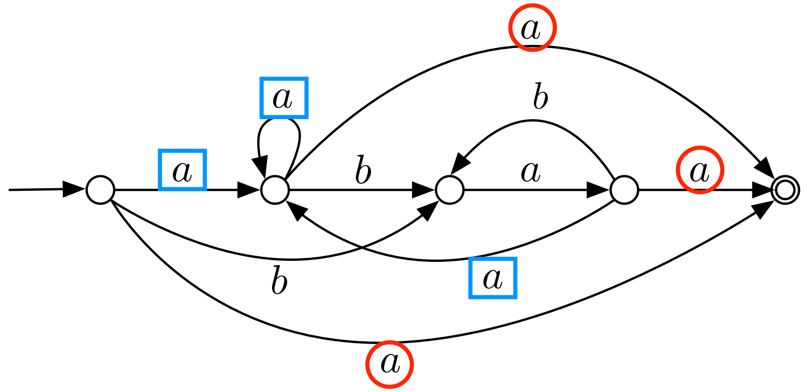
  - $|R|, |Q| = \Theta(Nd)$.
- An $O((nm)^{1-\epsilon})$ algorithm for regex matching => an $O((NM)^{1-\epsilon'})$ algorithm for OVP.
- Note that $|R| = m = \Theta(n)$.

Deterministic and Sparse Regular Expression Matching

Deterministic Regular Expressions

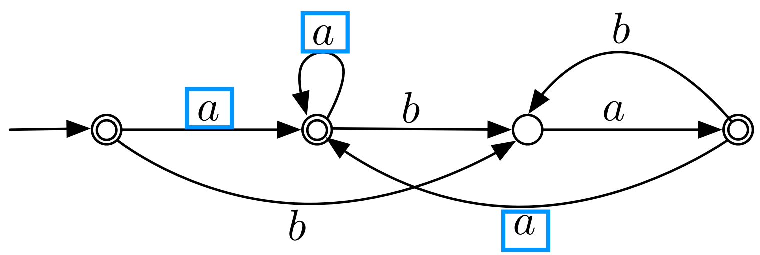
- Examples
  - Deterministic: $R_1 = (a \mid ba)^*$
  - Nondeterministic: $R_2 = (a \mid ba)^* a$
- Position automaton:

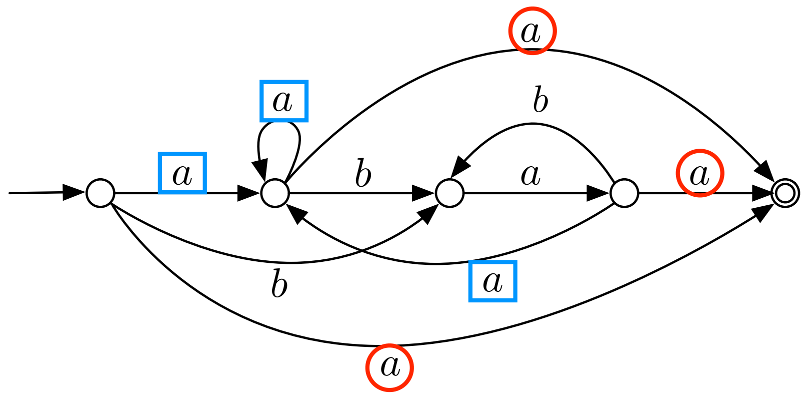




Deterministic Regular Expressions

- Examples
  - Deterministic: $R_1 = (a \mid ba)^*$
  - Nondeterministic: $R_2 = (a \mid ba)^* a$
- Position automaton:





- Groz, Maneth, and Staworko 2012:
  - Check if R is deterministic in linear time.
  - Matching in time $O(n \log \log m + m)$.
  - Other variants: k-ore in time $O(km)$.

Sparse Regular Expression Matching

- Sparsity measure [Bille and G.]:

  $\Delta_{R,Q}$ = #positions reached in position automation when matching Q with R.

- $1 \leq \Delta_{R,Q} \leq nm + 1$

- Matching in time $O(\Delta \log\log(nm/\Delta) + n + m)$ and space $O(m)$.

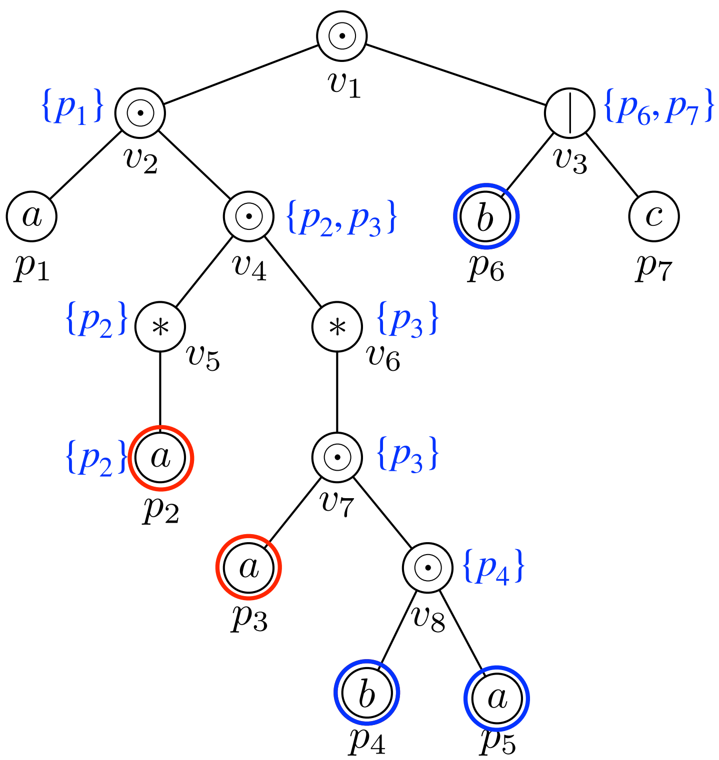
- Generalizes deterministic, k-ore.

- Never worse than $O(nm)$.

- Lower bound. For any $\Delta = n^{1+\gamma}$, where $\gamma \in (0,1]$ there exists no $O(\Delta^{1-\epsilon})$ algorithm for regular expression matching.

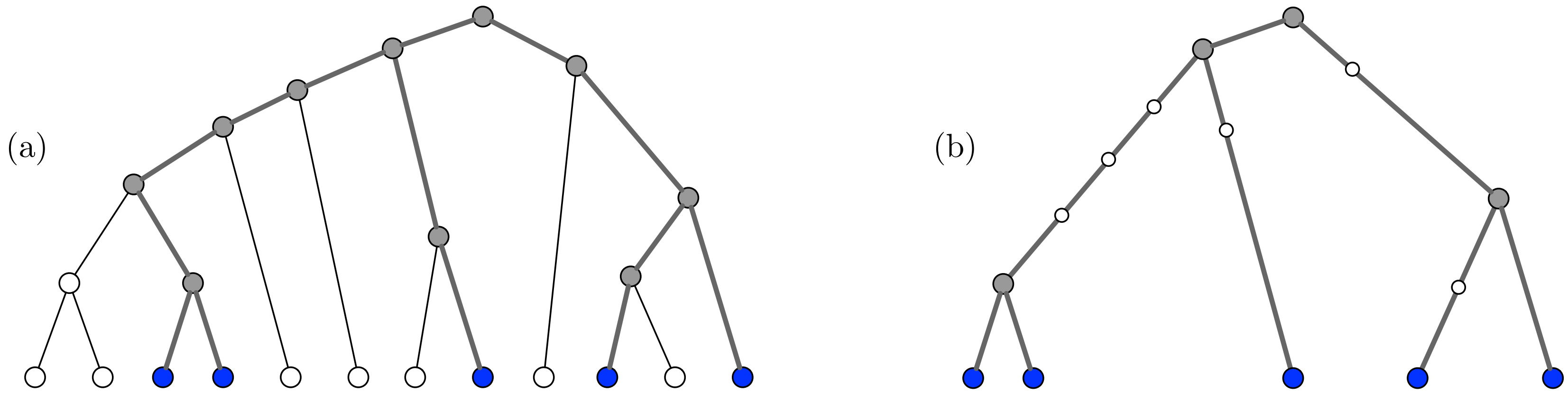
Techniques

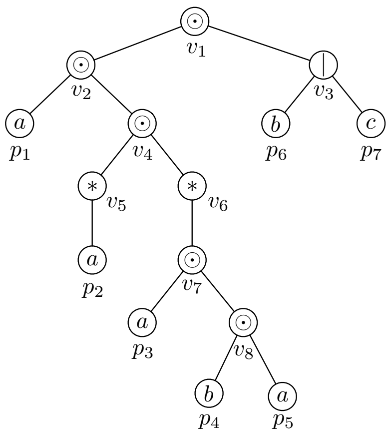
$$R = a(a^*)(aba)^*(b\,|\,c)$$



Techniques

$$R = a(a^*)(aba)^*(b \mid c)$$

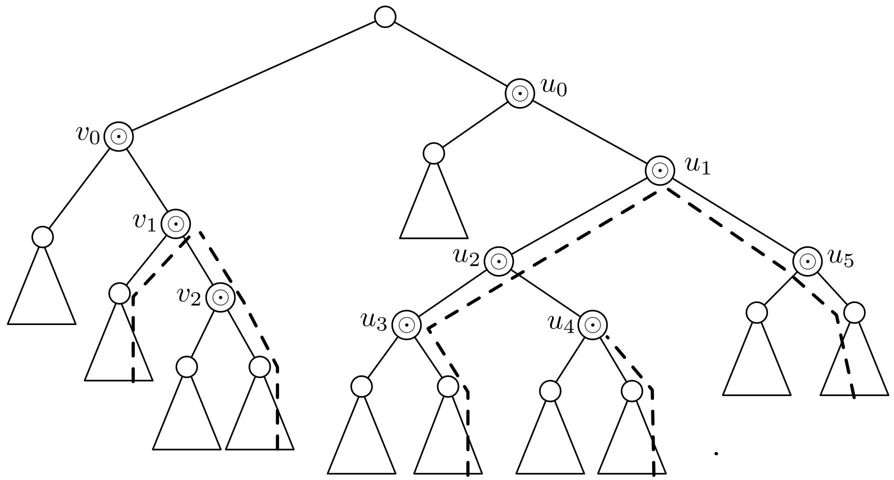




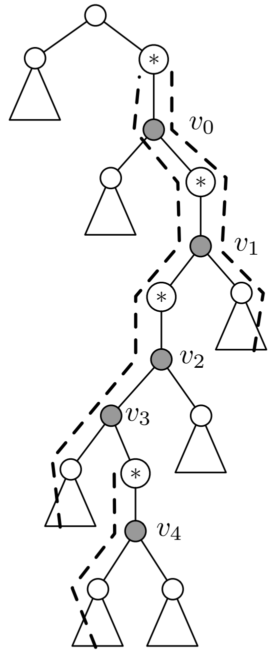
Techniques

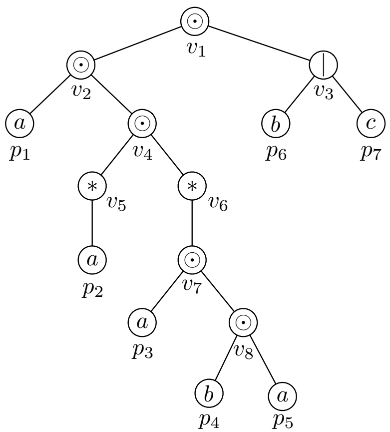
$$R = a(a^*)(aba)^*(b\,|\,c)$$

(a)

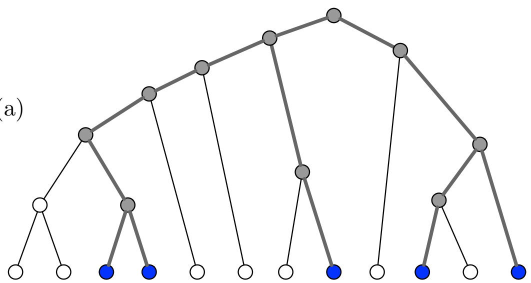


(b)

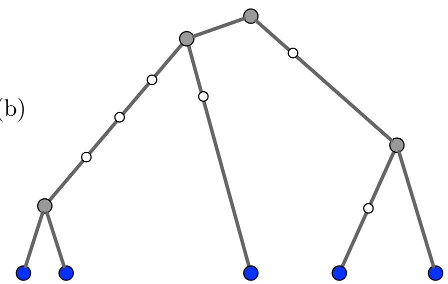




(a)



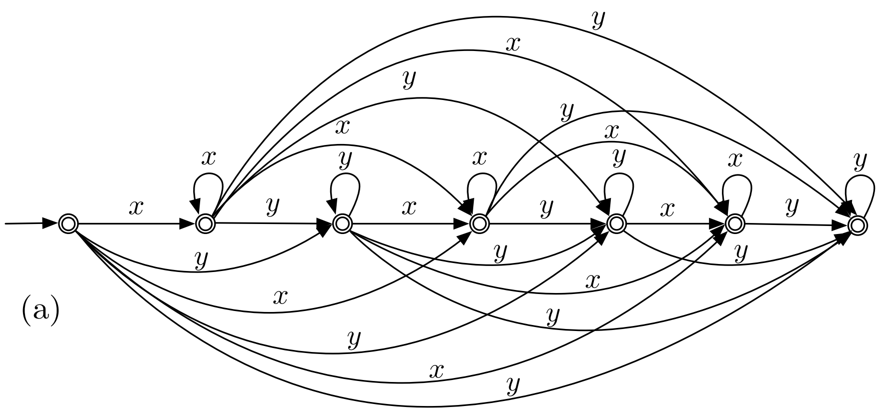
(b)



Lower bound

- Backurs and Indyk reduction:
  $\Delta = \Theta(nm)$.

$$R = \left(\bigcup_{j=1}^{|Q|} (x^* y^*) \right) \cdot P \cdot \left(\bigcup_{j=1}^{|Q|} (x^* y^*) \right)$$



(a)

- Our reduction:

$$R' = xxx\ldots xR$$
$$Q' = xxx\ldots xQ$$

- For any $\Delta = n^{1+\gamma}$, where $\gamma \in (0,1]$ there exists no $O(\Delta^{1-\epsilon})$ algorithm for regular expression matching.

Other parameterizations

- Words: $O((kn \log w)/w)$ where k = #number of words [Bille and Thorup 2010].
    - k vs Δ:
        - $\alpha_1 | \alpha_2 | \alpha_3 | \ldots | \alpha_m$ where $\alpha_i \neq \alpha_j$: $k = m$, $\Delta = 1$
        - ? $a{*}aaaaaa\ldots a$: $k = 2$, $\Delta = \Theta(mn)$ for $Q = aaaaaa\ldots a$
- Prefix sorting (Wheeler graphs special case):
    - $O(m^2 + np^2 \log(p \cdot \sigma))$ [Cotumaccio and Prezza 2021]
    - Wheeler graphs $O(m \log m + n \log \sigma)$ [Gagie, Manzini, Sirén 2017]
- Nondeterminism measures: cycle depth, width, tree width, path width, ….

Open problems

- Lower bounds in terms of words?
- More complicated expressions.