

XPath evaluation in linear time

Paweł Parys

Warsaw University

Joint work with Mikołaj Bojańczyk

Which fragment?

- navigation
- comparing data
(query $\alpha=\beta$, satisfied in nodes x such that some (x, y_1) is selected by α and some (x, y_2) is selected by β and data value in y_1 and y_2 is the same)

optional feature:

- counting
(query $count(\alpha)$, returning in each node the number of nodes y , such that (x, y) satisfies α)

Contributions

Input: XPath query Q , XML document D

Output: document tree nodes, which satisfy the query

The problem can be solved in complexity:

$O(|D| \cdot 4^{|Q|})$ – Bojańczyk, P (PODS 2008)

works also with counting queries

$O(|D| \cdot \log |D| \cdot |Q|^3)$

$O(|D| \cdot |Q|^3)$ – P (PODS 2009)

Very important subproblem

Initial input: Word w , regular language L

(Multiple) queries: positions i, j in w

answer: does the subword $w[i:j]$ belong to L ?

Complexity in $|w|$:

Preprocessing: $O(|w|) = k|w|$

Answering query: $O(1) = k$

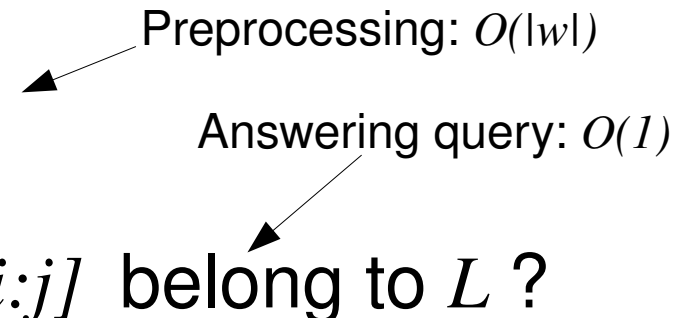
where k is the size of:

- monoid
- deterministic automaton
- nondeterministic automaton for special L
- nondeterministic automaton

PODS 2008

PODS 2008

Very important subproblem - relation to XPath

Initial input: Word w , regular language L 
(Multiple) queries: positions i, j in w
answer: does the subword $w[i:j]$ belong to L ?

Assume we have a word and each data value appears twice.

The above problem directly corresponds to a query $\text{self}=\beta$ with β describing L :

for each position i (and for j with the same data value)
we have to check if the subword $w[i:j] \in L$?

Very important subproblem - first solution

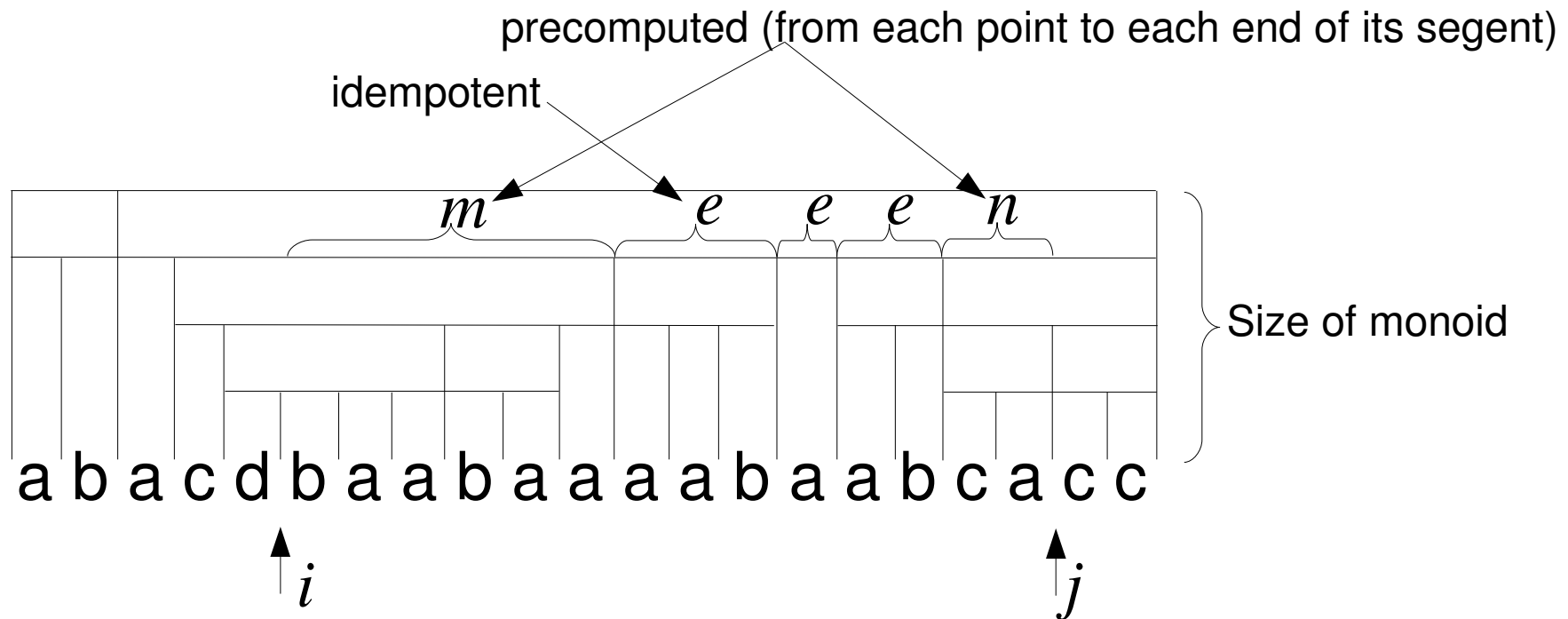
Initial input: Word w , regular language L

(Multiple) queries: positions i, j in w

answer: does the subword $w[i:j]$ belong to L ?

We use the Simon decomposition trees

(L is represented as a finite monoid - exponential blowup)



Very important subproblem - first solution

Initial input: Word w , regular language L

(Multiple) queries: positions i, j in w

answer: does the subword $w[i:j]$ belong to L ?

Another solution uses deterministic automata,
which also causes exponential blowup.

Very important subproblem - another solution

In real XPath there is no star in the path expressions, only multistep axes.

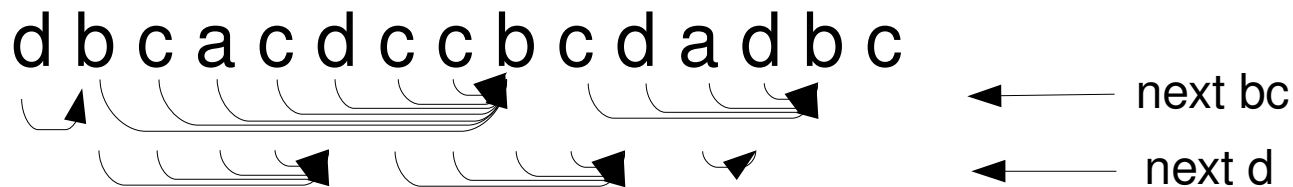
So we get special languages: star is allowed only around Σ .

For example: $a\Sigma^*bc\Sigma^*d\Sigma^*$

Initial input: Word w , language L of the above form

(Multiple) queries: positions i, j in w

answer: does the subword $w[i:j]$ belong to L ?



Better complexity: polynomial in L .

Very important subproblem - third solution

New work:

we want $k=|Q|^3$ for arbitrary nondeterministic automaton.

We use monoids implicitly, we work with the automaton.

Look at the Simon's decomposition tree:

however its height is $O(2^{|Q|})$, it has only $|w|$ nodes.

We use some weaker version of the decomposition trees.

Then the decomposition tree can be constructed in $O(|w| \cdot |Q|^3)$