

# XML Stream Processing

Dan Suciu

[www.cs.washington.edu/homes/suciu](http://www.cs.washington.edu/homes/suciu)

Joint work with faculty, visitors and students at UW

# Introduction

- This is a research project at UW
- Partially supported by MS
- Two parts:
  - A free toolkit of command lines: **xsort**, **xagg**, ...  
[www.cs.washington.edu/homes/suciu/XMLTK](http://www.cs.washington.edu/homes/suciu/XMLTK)
  - Research on XML stream processing – this talk

# The Problem

- Given:
  - Large number of Xpath expressions
  - Incoming stream of XML documents
- Decide for each document which expressions it matches

## XML Data Stream



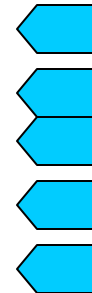
```
<datasets>
  <dataset>
  ...
</datasets>
```

## XPath expressions



```
/datasets/dataset
/datasets/dataset [history/text()="recent"]/title
/datasets/dataset //tableHead//*
/datasets/dataset //tableHead//*[text()="Galaxy"]
/datasets/dataset /history
/datasets/dataset /tableHead
/datasets/dataset /tableHead /field
```

## Decisions



# The Application(s)

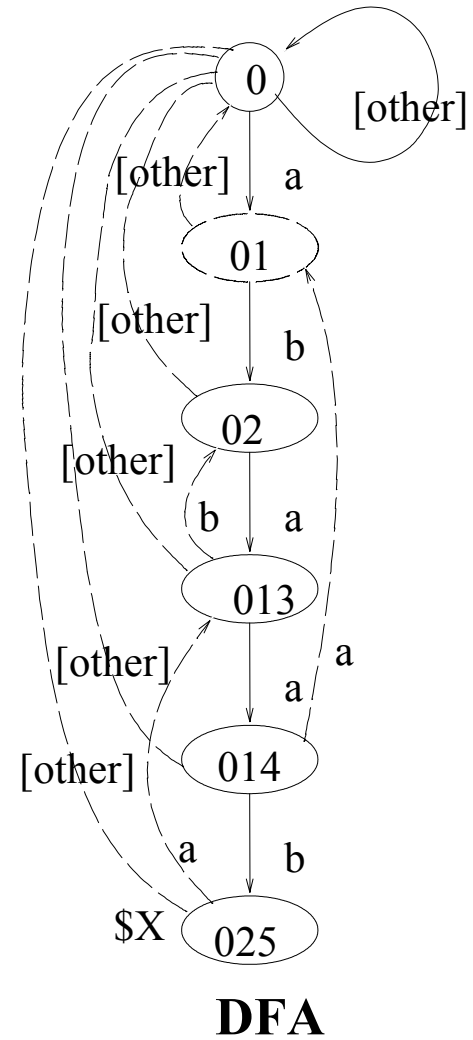
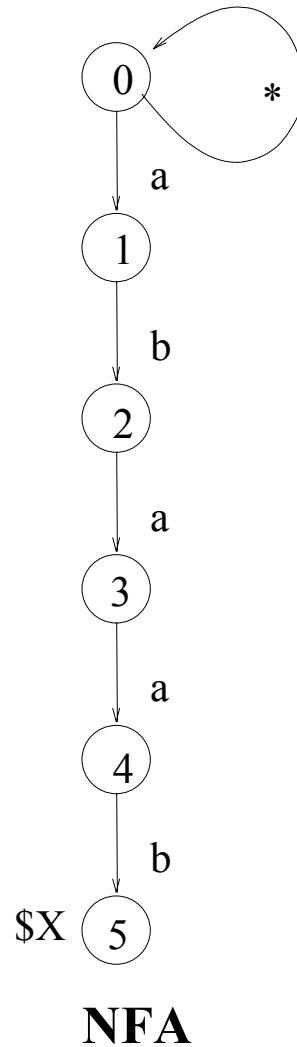
- Selective Dissemination of Information [Berkeley]
- XML content routing [MIT]
- SOAP Message routing in Application Servers
  
- Typical scale:
  - 10,000 to 1,000,000 Xpath expressions
  - XML stream: 1KB/s ? 1MB/s ?

# The Approaches

- Basic techniques
  - NFA plus optimizations: Xfilter/Yfilter, XTrie
  - DFA: we are doing this here
- Beyond the obvious
  - SIX
  - views

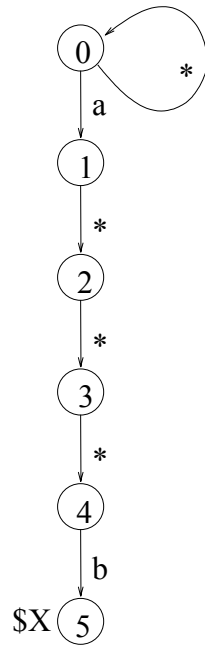
# Background on NFA and DFA

//a/b/a/a/b

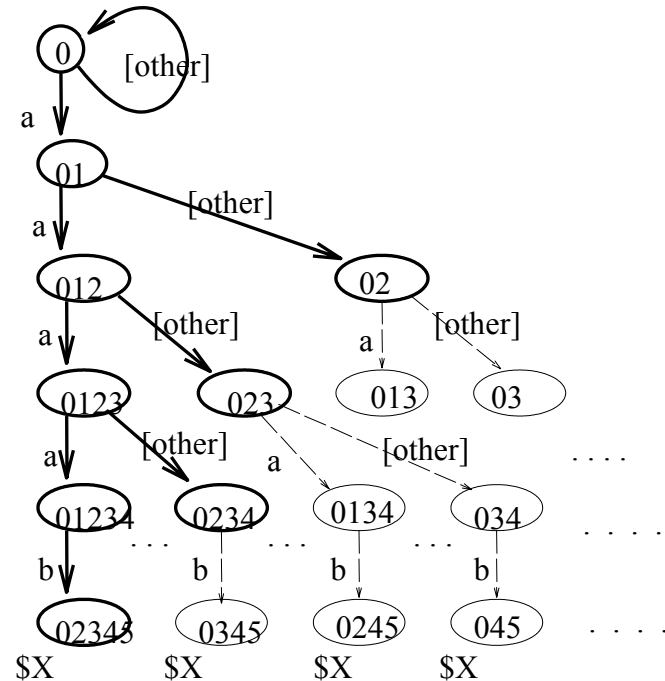


# Background on NFA and DFA

//a/\*\*/\*\*/\*\*b



NFA



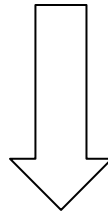
DFA (without back edges)

# Background on NFA and DFA

- Issue: need to linearize Xpath expressions

1 Xpath expression with filters

```
/catalog/product[@category="tools"][sales/@price > 200]/quantity
```



4 linear Xpath expressions

```
/catalog/product/$Y  
$Y/@category="tools"  
$Y/sales/@price  
$Y/quantity
```

Extra processing  
OK in trivial cases.  
Complex cases require  
more work (future)

For now: assume  
all Xpath expressions  
are linear



# Basic DFA Evaluation

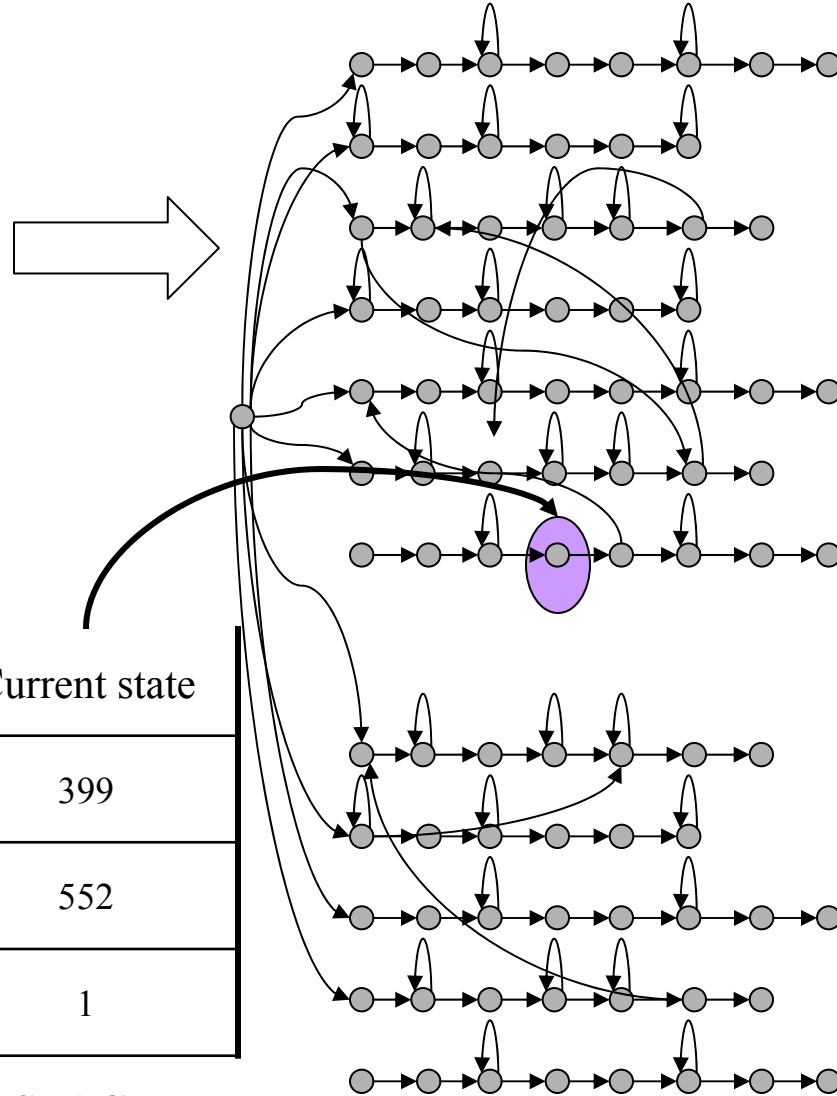
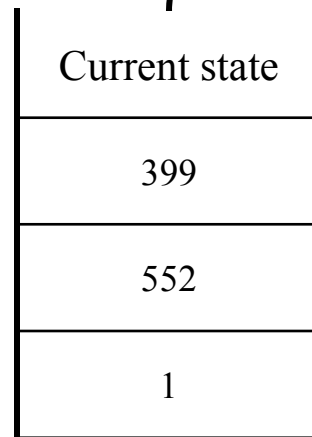
XPath

DFAs

```
/datasets/dataset  
/datasets/dataset [history/text()='recent']/title  
/datasets/dataset //tableHead//  
/datasets/dataset //tableHead//*/text()='Galaxy'  
/datasets/dataset /history  
/datasets/dataset /tableHead  
/datasets/dataset /tableHead /field  
/datasets/dataset  
/datasets/dataset [history/text()='recent']/title  
/datasets/dataset //tableHead//  
/datasets/dataset //tableHead//*/text()='Galaxy'  
/datasets/dataset /history  
/datasets/dataset /tableHead  
/datasets/dataset /tableHead /field  
...  
...  
...  
/datasets/dataset  
/datasets/dataset [history/text()='recent']/title  
/datasets/dataset //tableHead//  
/datasets/dataset //tableHead//*/text()='Galaxy'  
/datasets/dataset /history  
/datasets/dataset /tableHead  
/datasets/dataset /tableHead /field  
/datasets/dataset/history/text()
```

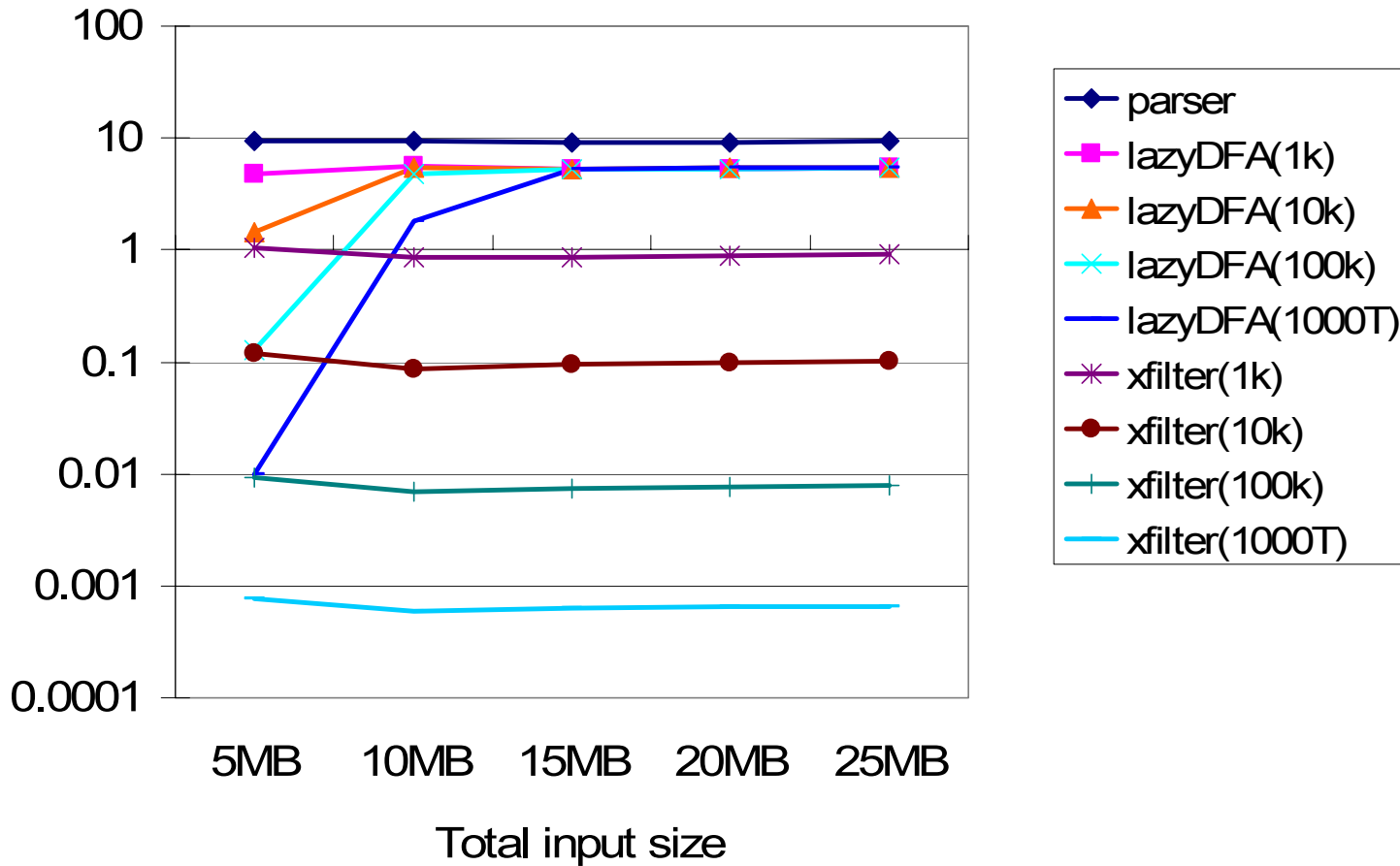
```
<datasets>  
  <dataset>  
  ...  
</datasets>
```

SAX  
events



# Comparison: Throughput in MB/s

Throughput for 1k, 10k, 100k, 1000k XPEs  
[ prob(\*)=10%, prob(//)=10% ]



# Number of States in DFA

Compute the DFA for 1,000,000 Xpath expressions ???!!?

- 1 linear Xpath → small DFA
- 1,000,000 linear Xpaths → HUGE DFA

# Number of States in DFA

**//section//footnote**

**//figure//footnote**

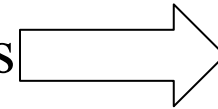
**//table//footnote**

....

....

**//abstract//footnote**

n Xpath expressions



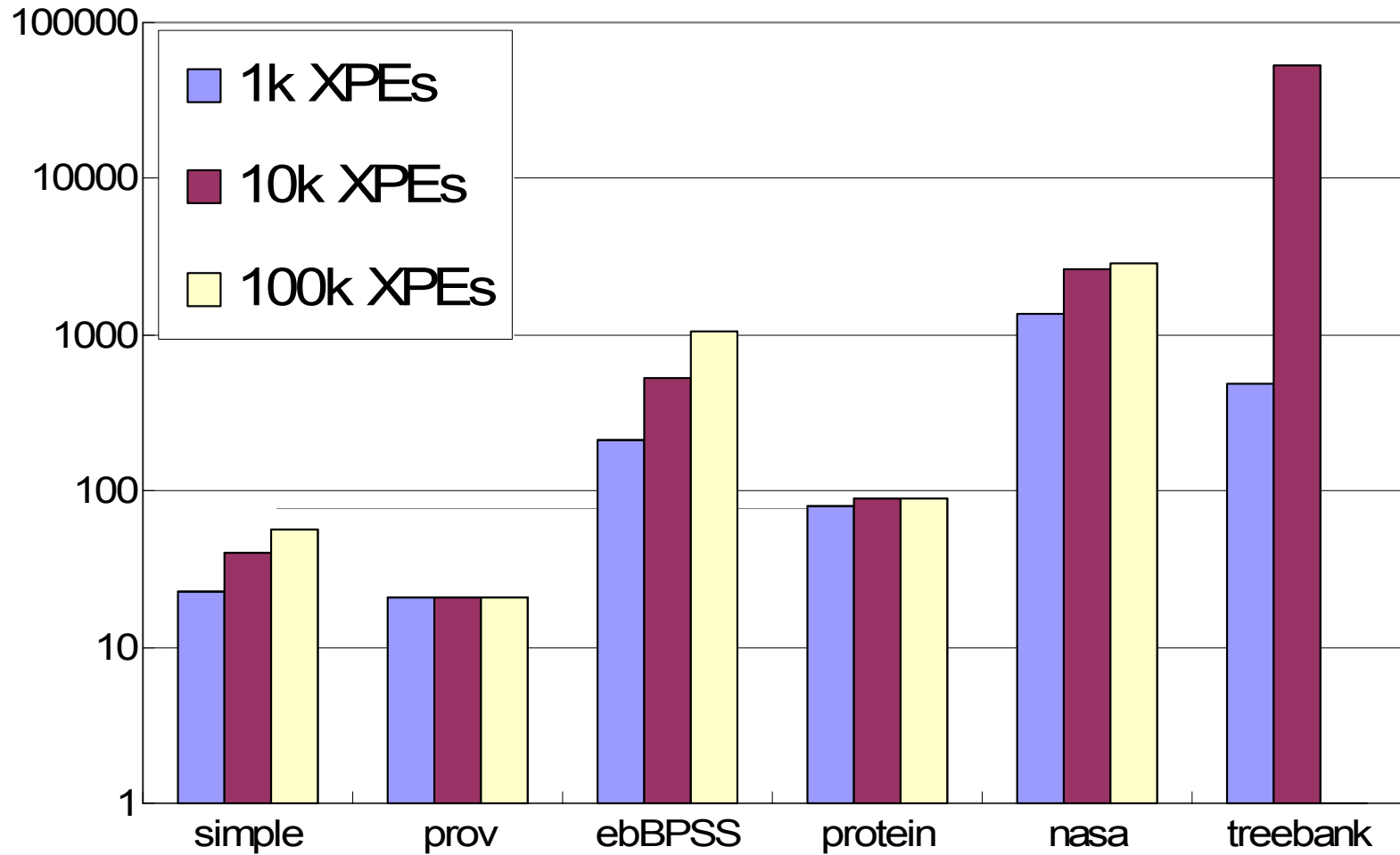
$2^n$  states

**Solution: lazy DFA !**

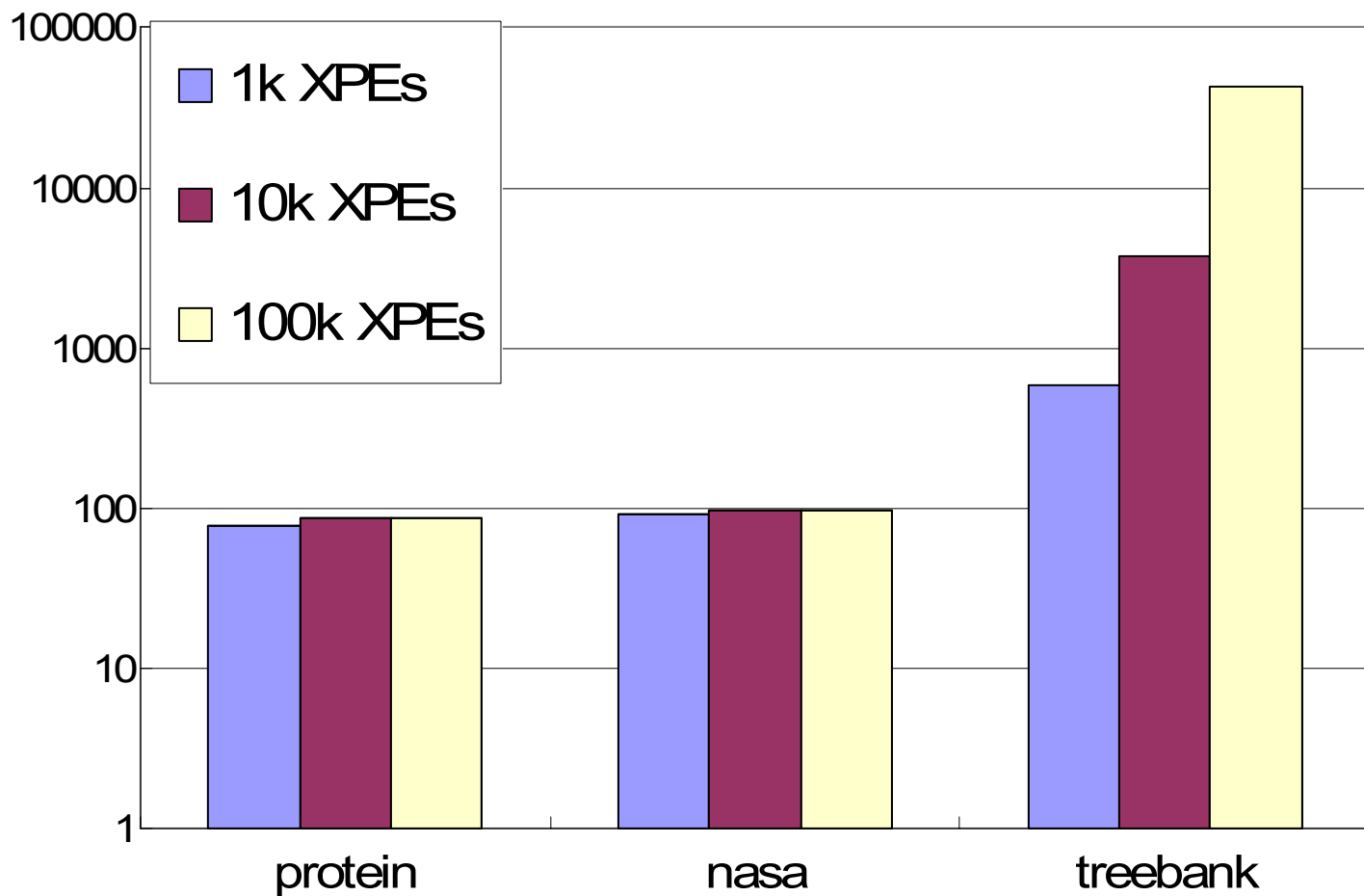
# Number of States in the lazy DFA

	Real XML data	Synthetic XML data
Non-recursive or data-style recursive DTDs	<b>Theorem</b> DFA is small	<b>Theorem</b> DFA is small
Document-style recursive DTD	<b>Theorem</b> DFA is small	DFA is HUGE

## Number of DFA States - SYNTHETIC Data



Number of DFA States - REAL Data



# Beyond the Obvious I: Stream IndeX (SIX)

Main observation:

- Parsing is major bottleneck
- Skip portions of the XML document → avoid parsing and processing

# Stream Index (SIX)

## XML

```
<bib>
  <book> <publisher> Addison-Wesley </publisher>
    <author> Serge Abiteboul </author>
    <author> <first-name> Rick </first-name>
      <last-name> Hull </last-name>
    </author>
    <author> Victor Vianu </author>
    <title> Foundations of Databases </title>
    <year> 1995 </year>
  </book>
  <book price="55">
    <publisher> Freeman </publisher>
    <author> Jeffrey D. Ullman </author>
    <title> Principles of Database and
      Knowledge Base Systems </title>
    <year> 1998 </year>
  </book>
</bib>
```

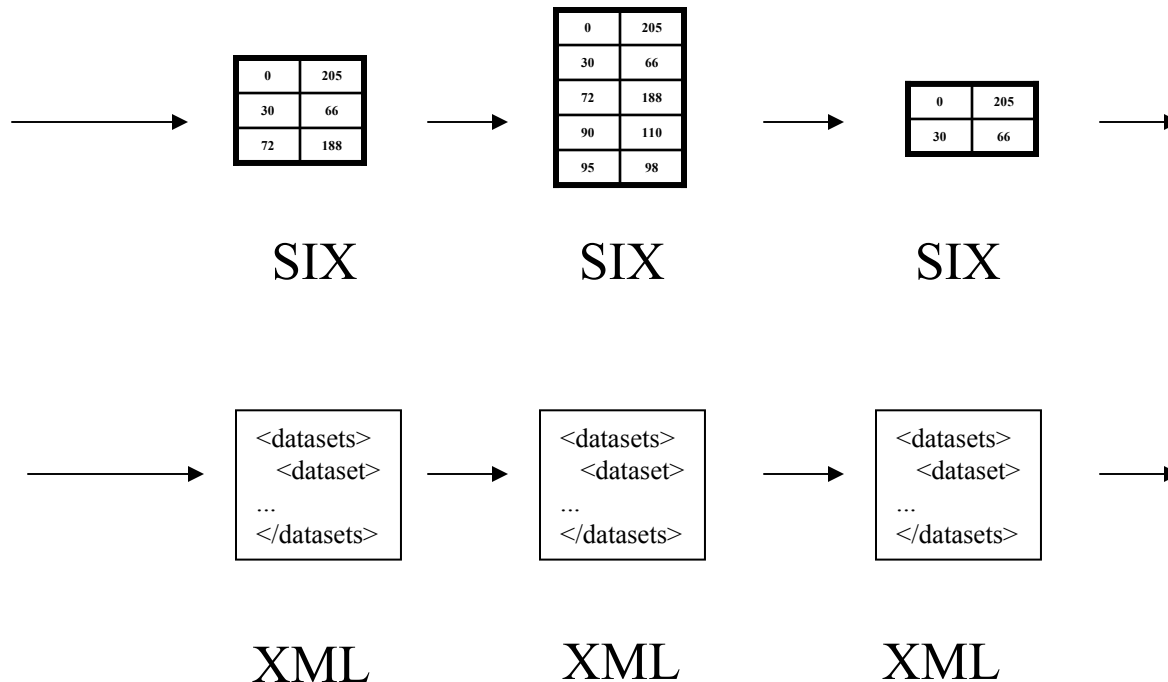
## SIX

	<b>beginOffset</b>	<b>endOffset</b>
<b>bib</b>	0	1490124
<b>book</b>	3	409023
<b>publisher</b>	12	423
<b>author</b>	426	879
<b>author</b>	978	...
...		

# Stream Index (SIX)

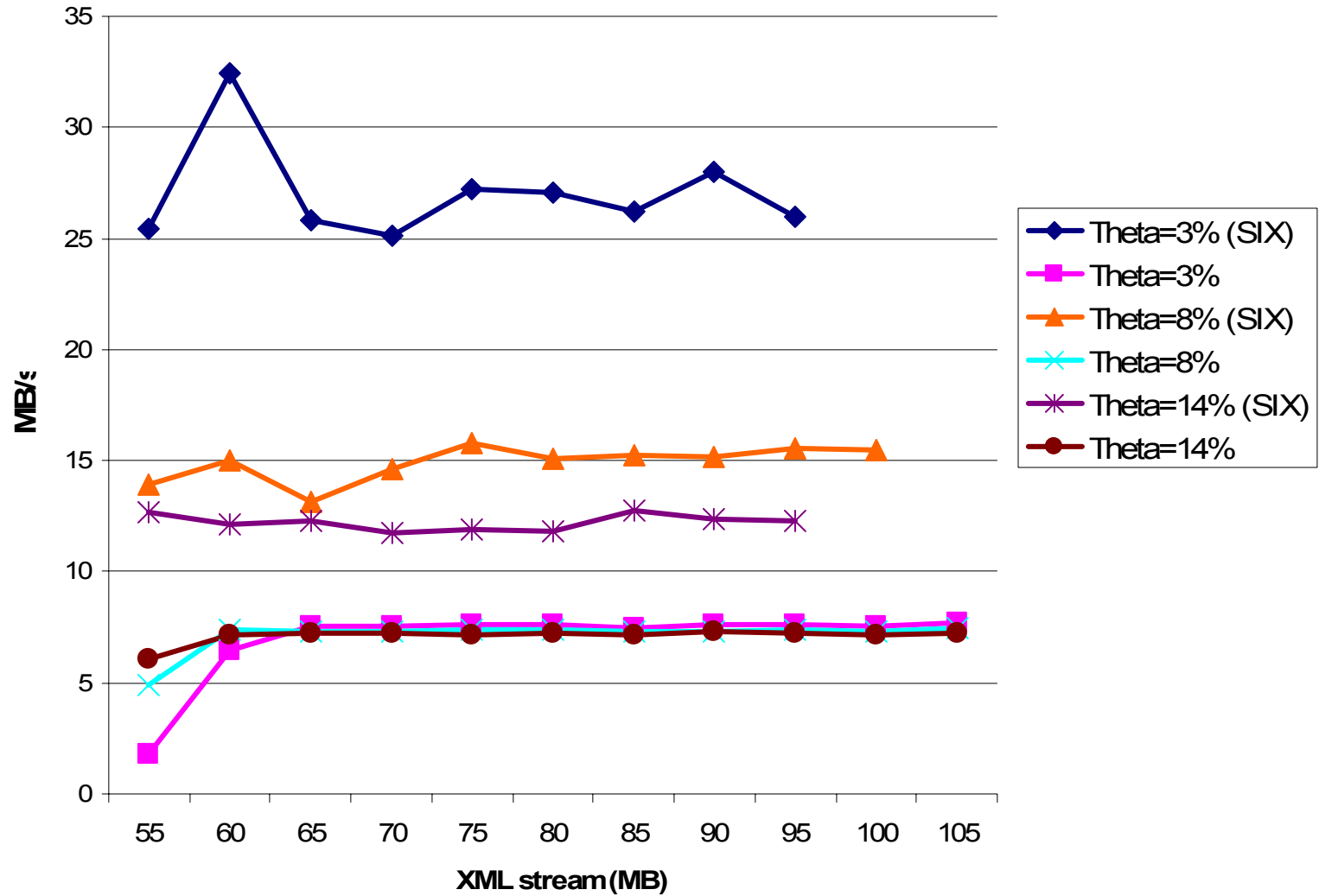
- API for SIX:
  - **skip(k)**, where  $k \geq 0$
  - skips to the end of the k'th surrounding element
  - Uses **beginOffset** to sync with the XML doc
  - Uses **endOffset** to skip

# Stream Index (SIX)



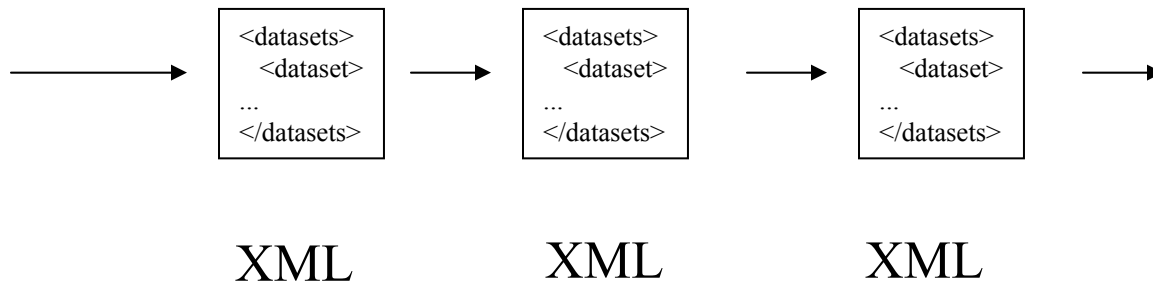
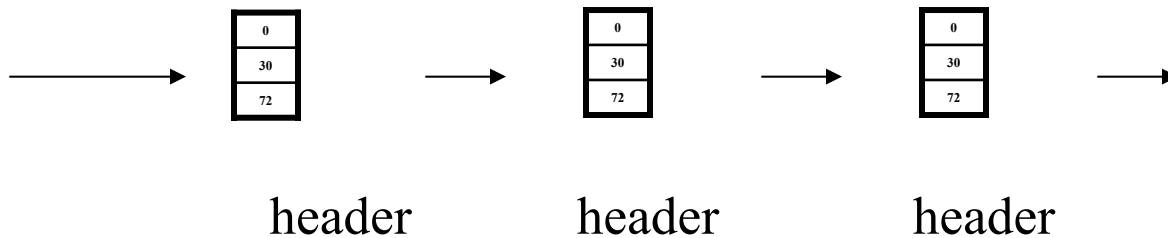
The SIX stream is about 6% of the data stream  
And can be made MUCH smaller

### Throughput improvements from SIX (stable)



# Beyond the obvious II: View Selections

- On-going work: View selections → header



100x speedup  
On a hit

# Conclusions

- Two ideas:
  - Computing the DFA is possible !
  - Use extra info to further speedup: SIX, Headers
- Issues:
  - Extend DFAs to filters: process *events*
  - How to represent SIX or Headers in XML

- [Msdn.microsoft.com/webservices](https://msdn.microsoft.com/webservices)
- [Steveluc@microsoft.com](mailto:Steveluc@microsoft.com)      contact