Tailor-Made Native XML Storage Structures

Karsten Schmidt Theo Härder
{kschmidt|haerder}@informatik.uni-kl.de

AG DBIS
University of Kaiserslautern, Germany

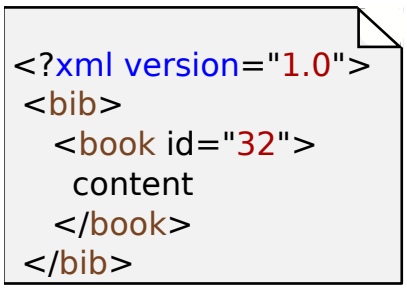
Conference on
"Advances in Databases and Information Systems",
Varna, Bulgaria

DASMOD
Dependable Adaptive Systems and Mathematical Modeling

Outline

XML Databases

Current Situation



XML Databases

Current Situation

```
<?xml version="1.0">
 <bib>
   <book id="32">
    content
   </book>
 </bib>
```

shredding

ID	TAG	CONTENT	PARENT
1	bib	–	–
2	book	–	1
3	id	32	2
n	...	–	...
n	...	–	...

NT

XML Databases

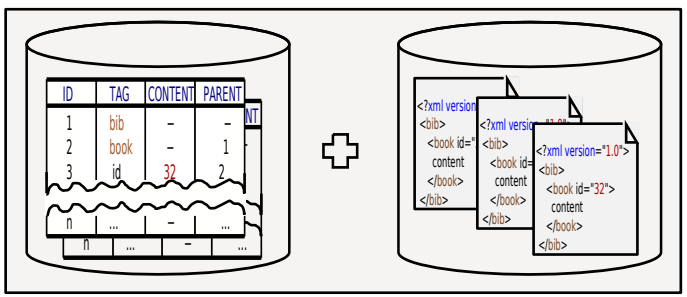
Current Situation

```
<?xml version="1.0">
 <bib>
   <book id="32">
    content
   </book>
 </bib>
```

shredding

ID	TAG	CONTENT	PARENT
1	bib	–	–
2	book	–	1
3	id	32	2
n	...	–	...
n	...	–	...

extension



XML Databases

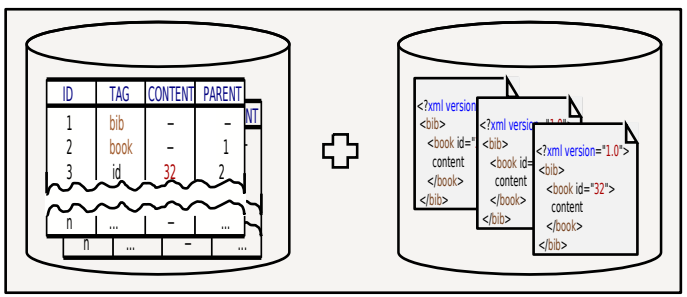
Current Situation

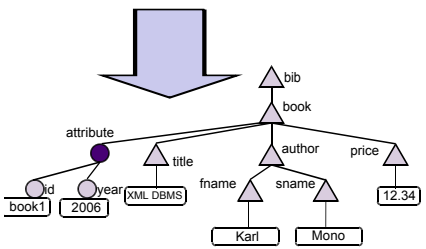
```
<?xml version="1.0">
 <bib>
   <book id="32">
    content
   </book>
 </bib>
```

shredding

ID	TAG	CONTENT	PARENT
1	bib	–	–
2	book	–	1
3	id	32	2
n	...	–	...
n	...	–	...

extension





XML Databases

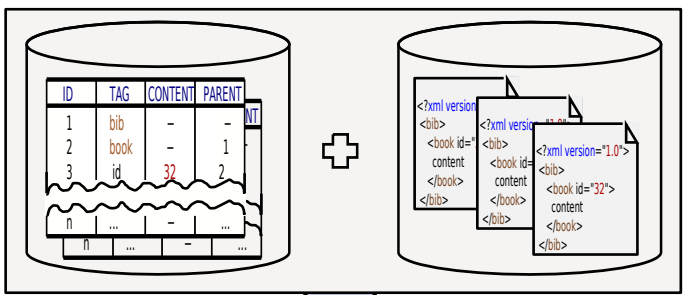
Current Situation

```
<?xml version="1.0">
 <bib>
   <book id="32">
    content
   </book>
 </bib>
```

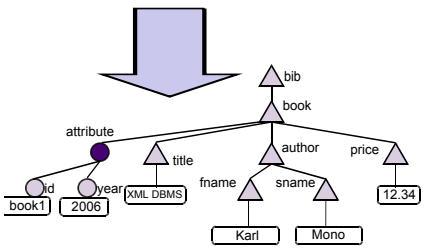
shredding

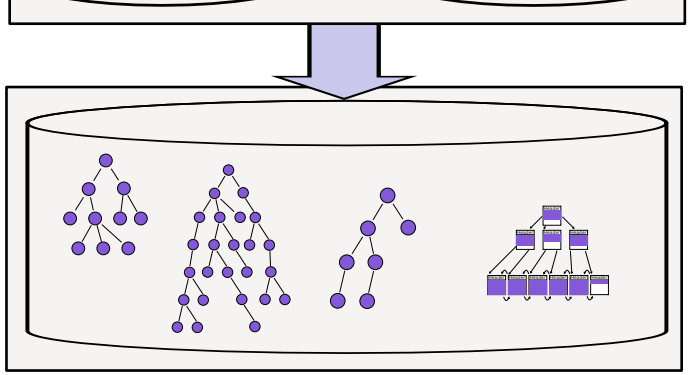
ID	TAG	CONTENT	PARENT
1	bib	–	–
2	book	–	1
3	id	32	2
n	...	–	...
n	...	–	...

extension



native





XML Databases

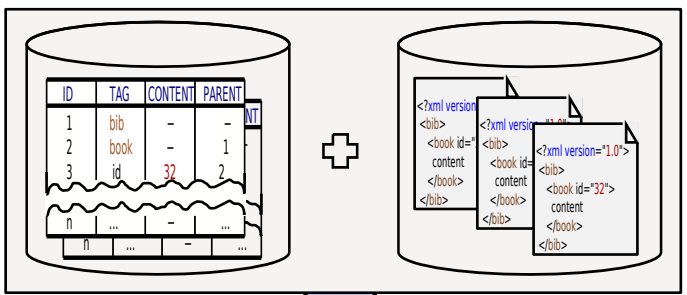
Current Situation

```
<?xml version="1.0">
 <bib>
   <book id="32">
    content
   </book>
 </bib>
```

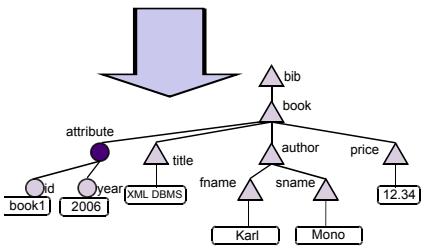
shredding

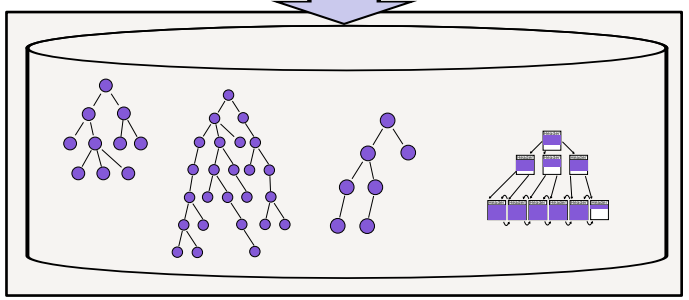
ID	TAG	CONTENT	PARENT
1	bib	–	–
2	book	–	1
3	id	32	2
n	...	–	...
n	...	–	...

extension



native





PROBLEMS CHALLENGES & OPEN ISSUES ??

Round Trip property & Variety of XML documents
(performance, indexing, access −operators / structures, ...)

Storage Structures

XML document

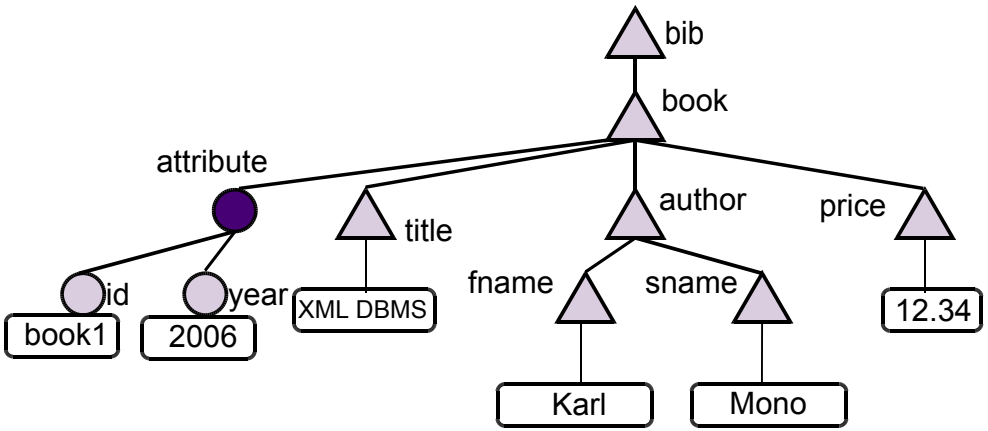
```
<?xml version="1.0"?>
<bib>
  <book year="2006"
        id="book1">
    <title>XML DBMS</title>
    <author>
      <fname>Karl</fname>
      <sname>Mono</sname>
    </author>
    <price>12.34</price>
  </book>
</bib>
```

Storage Structures - Tree View

XML document

```
<?xml version="1.0"?>
<bib>
  <book year="2006"
        id="book1">
    <title>XML DBMS</title>
    <author>
      <fname>Karl</fname>
      <sname>Mono</sname>
    </author>
    <price>12.34</price>
  </book>
</bib>
```

Tree View

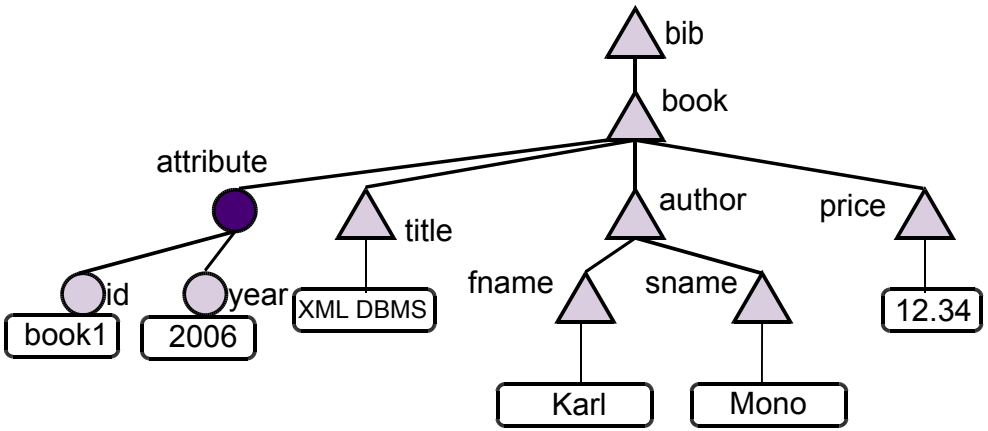


Storage Structures - Tree View

XML document

```
<?xml version="1.0"?>
<bib>
  <book year="2006"
        id="book1">
    <title>XML DBMS</title>
    <author>
      <fname>Karl</fname>
      <sname>Mono</sname>
    </author>
    <price>12.34</price>
  </book>
</bib>
```

Tree View



- Variable tree width/depth
- Separation into element nodes, text nodes and attribute nodes
- Determination of paths, inner nodes and leaves

Storage Structures - Node Labeling

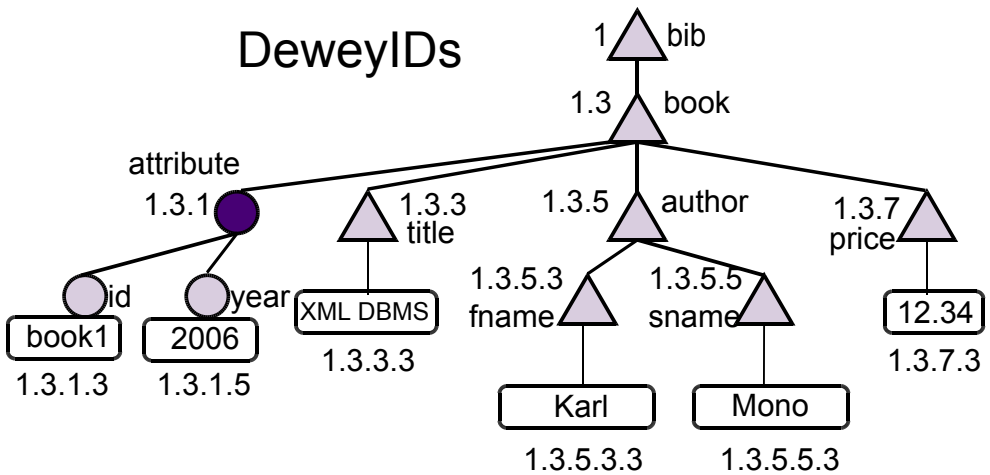
Properties

- Dynamic assignment
- Preserve order
- Stable on modifications
- Space consumption
- Query support

⇒ Prefix-based

  - Support for XPath axes, locks, and compression

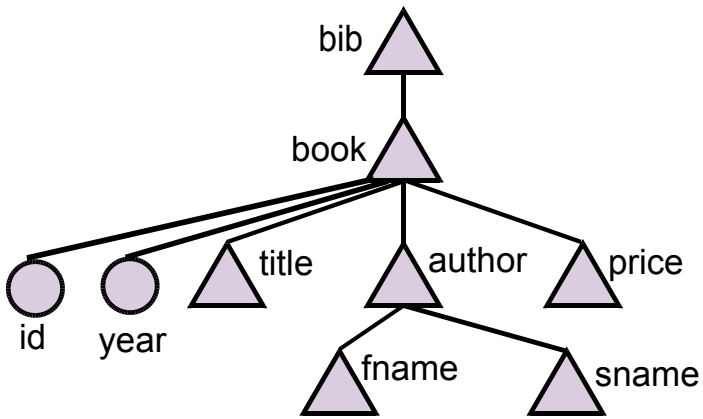
- e.g.: OrdPath, DeweyID, DLN



Path Synopsis - Improving Storage Consumption

Path Synopsis

- Concise representation of path classes
- Basis for the “elementless” storage



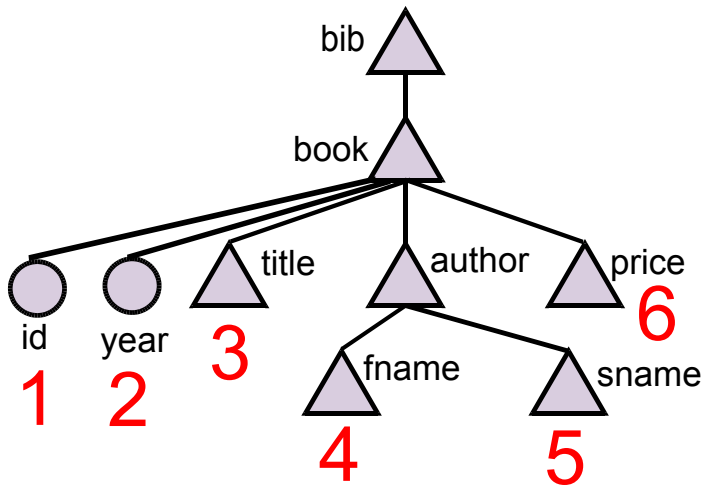
Path Synopsis - Improving Storage Consumption

Path Synopsis

- Concise representation of path classes
- Basis for the "elementless" storage

Path Class Reference - PCRs

- **1** /bib/book@id
- **2** /bib/book@year
- ...
- **6** /bib/book/price



Content Compression - Improving Storage Consumption

Requirements

- Compression ratio vs. speed
- Full document flexibility (indexes, query processing, modifications) $\longrightarrow$ disqualifies XMill, XGRIND, Xpress, ...

Content Compression - Improving Storage Consumption

Requirements

- Compression ratio vs. speed
- Full document flexibility (indexes, query processing, modifications) $\longrightarrow$ disqualifies XMill, XGRIND, Xpress, ...

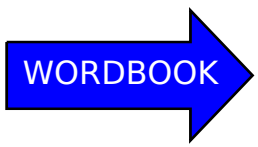
```
<?xml version="1.0">
 <bib>
   <book id="32">
     XML documents in databases
     ... suitable storage model of XML
     ... databases for documents in ...
   </book>
 </bib>
```

Content Compression - Improving Storage Consumption

Requirements

- Compression ratio vs. speed
- Full document flexibility (indexes, query processing, modifications) $\longrightarrow$ disqualifies XMill, XGRIND, Xpress, ...

```
<?xml version="1.0">
 <bib>
   <book id="32">
     XML documents in databases
     ... suitable storage model of XML
     ... databases for documents in ...
   </book>
 </bib>
```

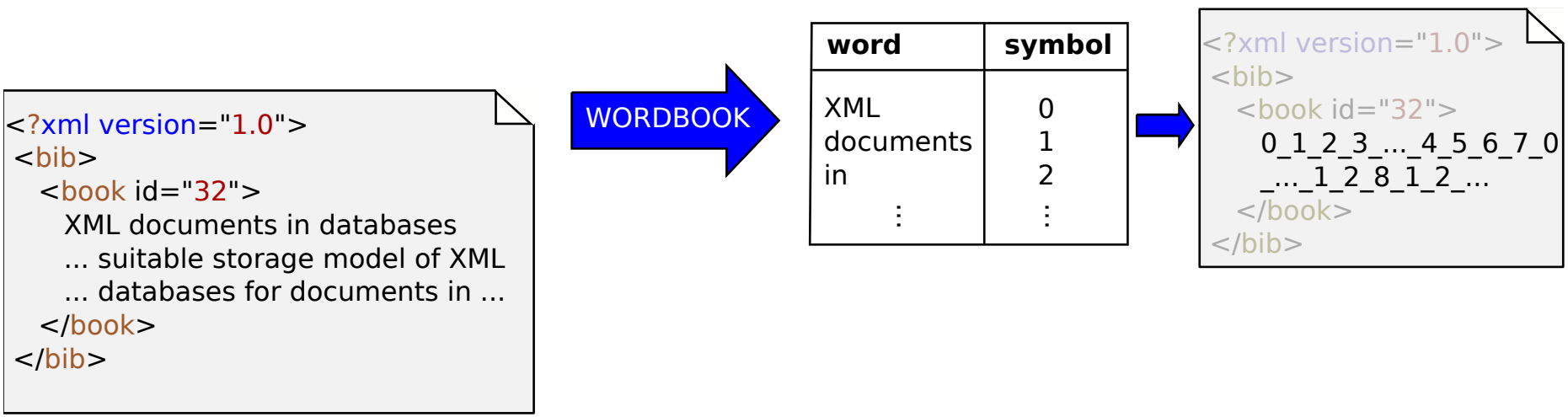


word	symbol
XML	0
documents	1
in	2
⋮	⋮

Content Compression - Improving Storage Consumption

Requirements

- Compression ratio vs. speed
- Full document flexibility (indexes, query processing, modifications) $\rightarrow$ disqualifies XMill, XGRIND, Xpress, ...

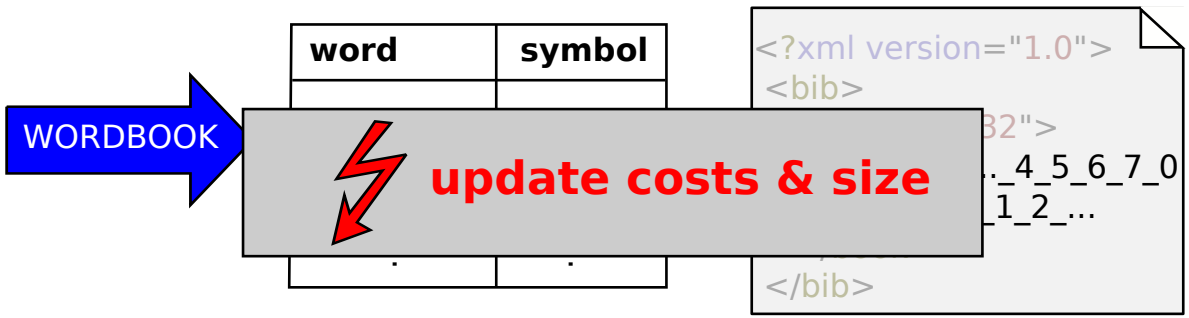


Content Compression - Improving Storage Consumption

Requirements

- Compression ratio vs. speed
- Full document flexibility (indexes, query processing, modifications) $\longrightarrow$ disqualifies XMill, XGRIND, Xpress, ...

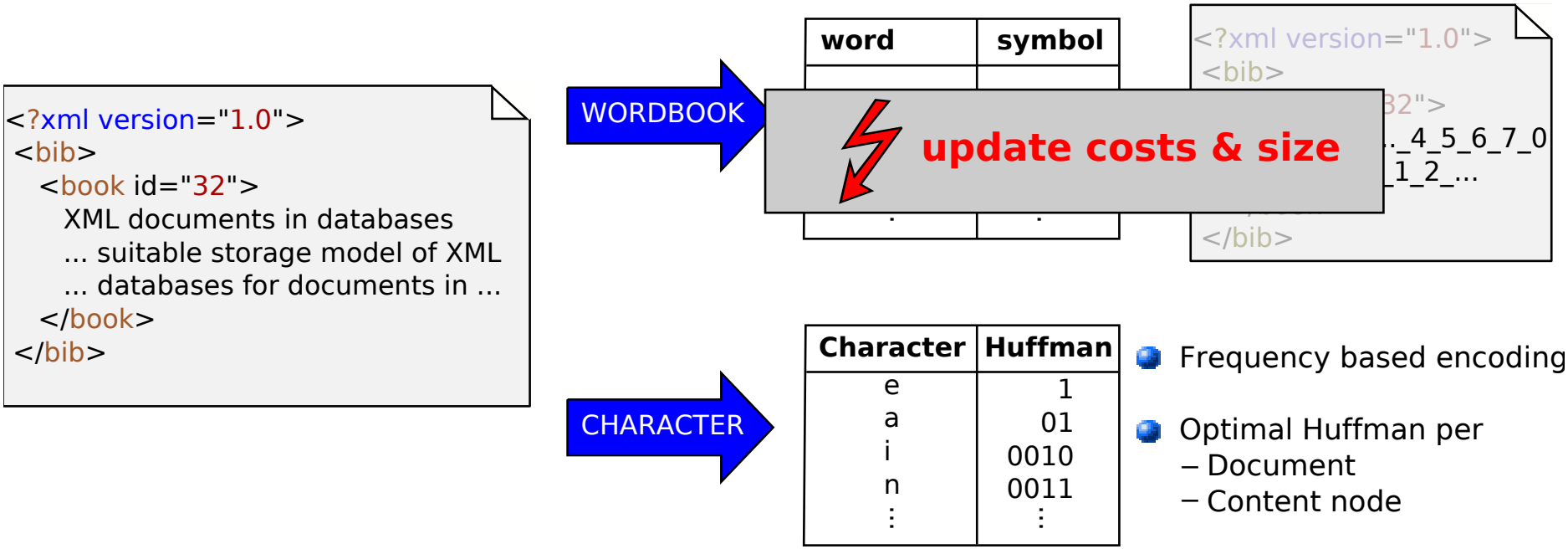
```
<?xml version="1.0">
<bib>
  <book id="32">
    XML documents in databases
    ... suitable storage model of XML
    ... databases for documents in ...
  </book>
</bib>
```



Content Compression - Improving Storage Consumption

Requirements

- Compression ratio vs. speed
- Full document flexibility (indexes, query processing, modifications) $\longrightarrow$ disqualifies XMill, XGRIND, Xpress, ...

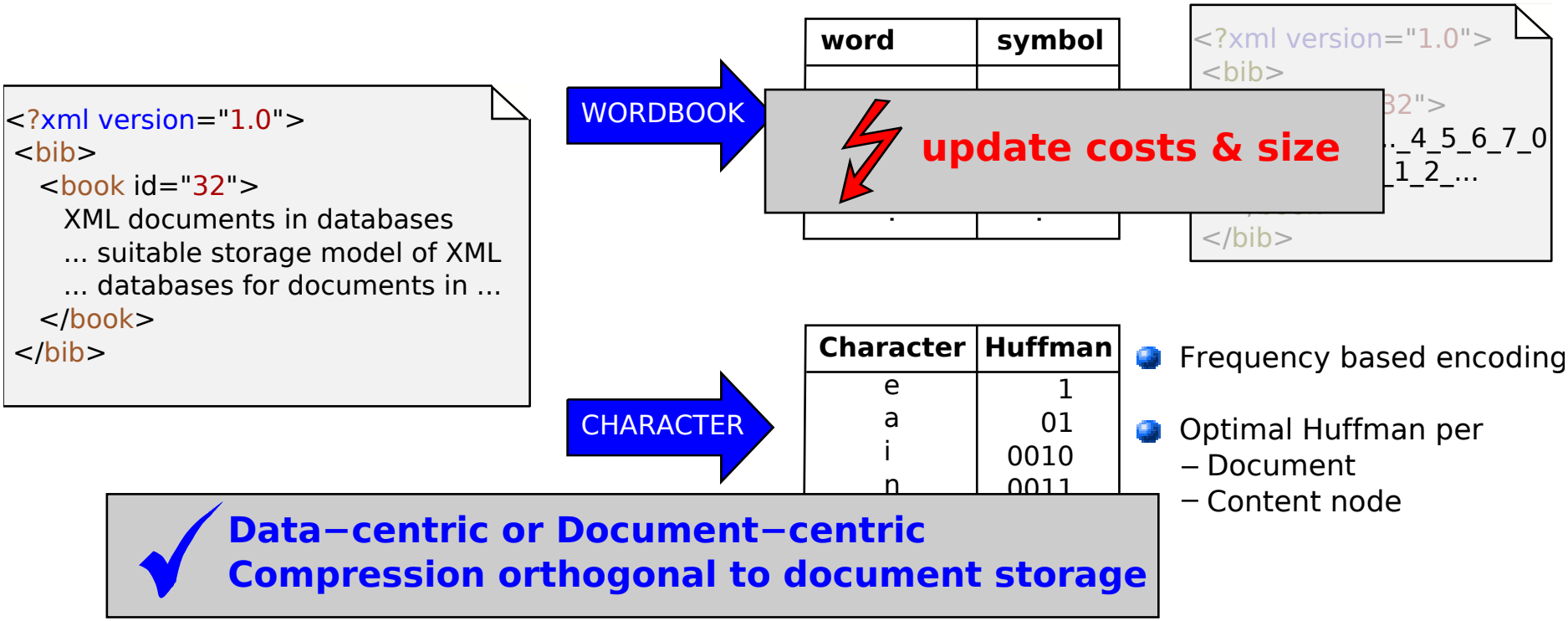


- Frequency based encoding
- Optimal Huffman per
  - Document
  - Content node

Content Compression - Improving Storage Consumption

Requirements

- Compression ratio vs. speed
- Full document flexibility (indexes, query processing, modifications) → disqualifies XMill, XGRIND, Xpress, ...



Example Documents & Variety

1. Size

Example Documents & Variety

1. Size
2. No of Attributes & Elements

Example Documents & Variety

1. Size
2. No of Attributes & Elements
3. No of Text Nodes

Example Documents & Variety

1. Size
2. No of Attributes & Elements
3. No of Text Nodes
4. No of Path Classes

Example Documents & Variety

1. Size
2. No of Attributes & Elements
3. No of Text Nodes
4. No of Path Classes
5. Max. Depth / Avg. Depth

Example Documents & Variety

1. Size — 32 MB - 1,820 MB
2. No of Attributes & Elements — 1 Mio - 82 Mio
3. No of Text Nodes — 1 Mio - 54 Mio
4. No of Path Classes — 17 - 220,894
5. Max. Depth / Avg. Depth — 4/3.3 - 37/8.4

- Example documents: lineitem uniprot dblp treebank

Example Documents & Variety

1. Size — 32 MB - 1,820 MB
2. No of Attributes & Elements — 1 Mio - 82 Mio
3. No of Text Nodes — 1 Mio - 54 Mio
4. No of Path Classes — 17 - 220,894
5. Max. Depth / Avg. Depth — 4/3.3 - 37/8.4

- Widespread document parameters: factor 10^4 and more!
- Simple structured vs. complex documents
- Storage parameters need adjustment

(Pre)-Analysis?

Analysis and Sampling (Step 1)

Purpose

- Collect XML document specific characteristics (elements, text length/nodes, attributes, path classes, vocabulary, depth, fan-out, ...)
- Optimize physical database parameters (page layout, record format, Dewey encoding, storage structures, indexes, ...)

Analysis and Sampling (Step 1)

Purpose

- Collect XML document specific characteristics (elements, text length/nodes, attributes, path classes, vocabulary, depth, fan-out, ...)
- Optimize physical database parameters (page layout, record format, Dewey encoding, storage structures, indexes, ...)

Complete Pre-Analysis

– Scans the complete document
\+ Precise parametrization
\+ Predictable space consumption
– Long runtime

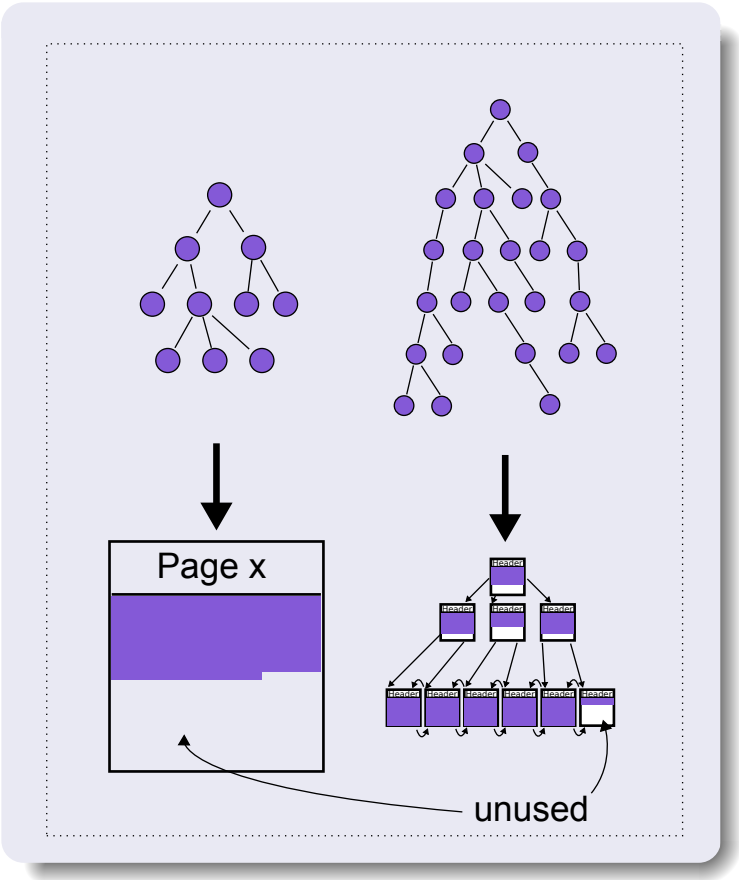
Sampling

– Scans the first x bytes
\+ Short runtime
\+ Stable parameters for $< 10\%$
\+ By buffering XML streams analyzable
– Limited to document head
– Imprecise parametrization
– Worst case treatment necessary

Configuration (Step 2)

Model 1 - Single Documents

- Dedicated storage per document
- Optional document index (big documents)
- Additional indexes/path synopsis separated

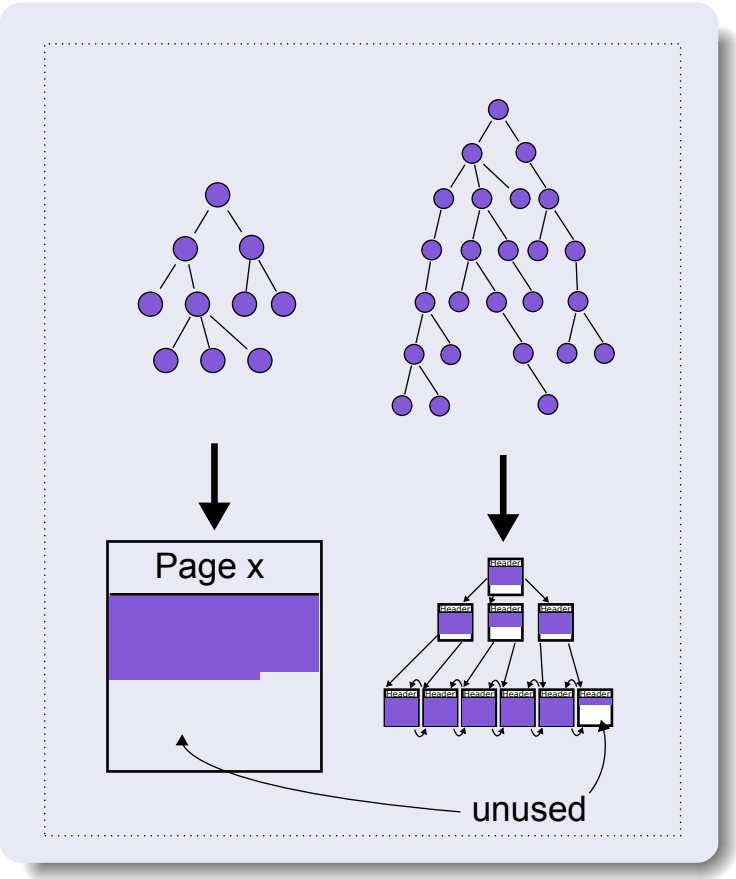


Configuration (Step 2)

Model 1 - Single Documents

- Dedicated storage per document
- Optional document index (big documents)
- Additional indexes/path synopsis separated

- Single (big) documents
- Sequential scan (SAX)
- Transactional processing (DeweyID)
- Structure independence



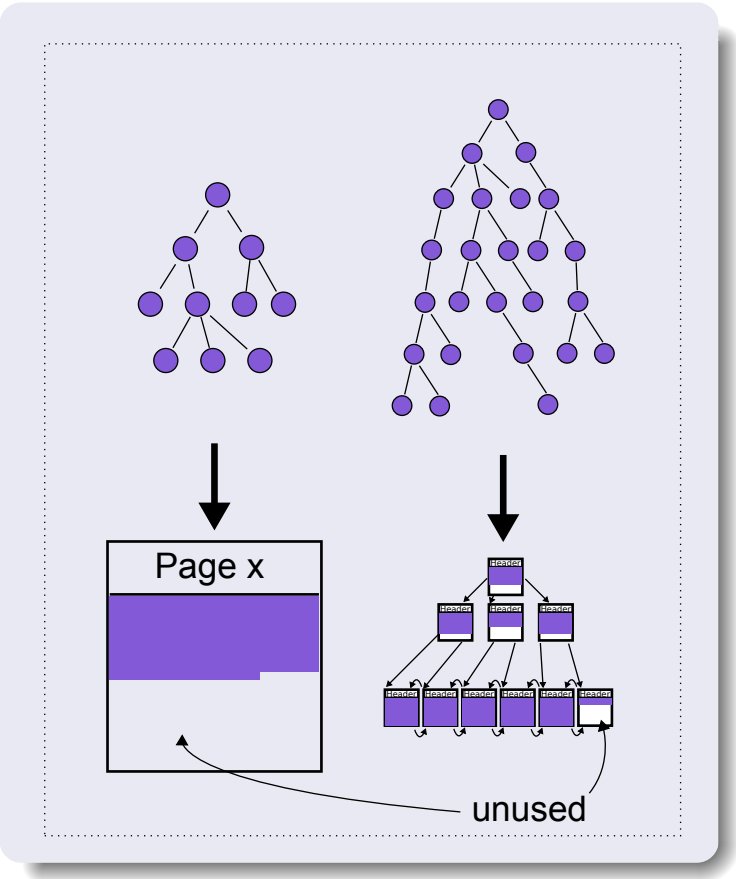
Configuration (Step 2)

Model 1 - Single Documents

- Dedicated storage per document
- Optional document index (big documents)
- Additional indexes/path synopsis separated

- Single (big) documents
- Sequential scan (SAX)
- Transactional processing (DeweyID)
- Structure independence

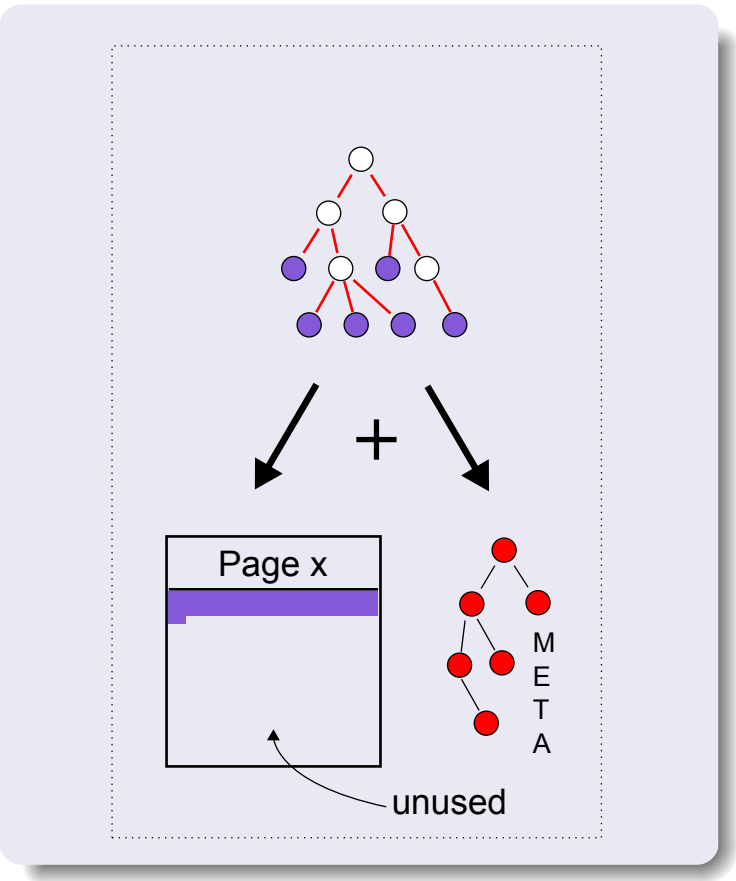
- Unused space
  document size $<$ page size



Configuration (Step 2)

Model 2 - "Elementless"

- Dedicated storage
- Indexes optional
- Path synopsis (PCRs) mandatory
- Solely leaf nodes stored
- Inner structure rebuilt on-demand

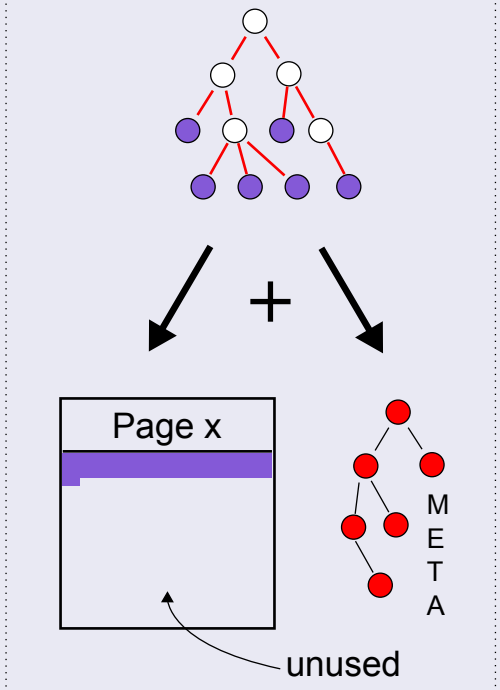


Configuration (Step 2)

Model 2 - "Elementless"

- Dedicated storage
- Indexes optional
- Path synopsis (PCRs) mandatory
- Solely leaf nodes stored
- Inner structure rebuilt on-demand

- Reduced redundancy reduces storage space
- Retain all document properties



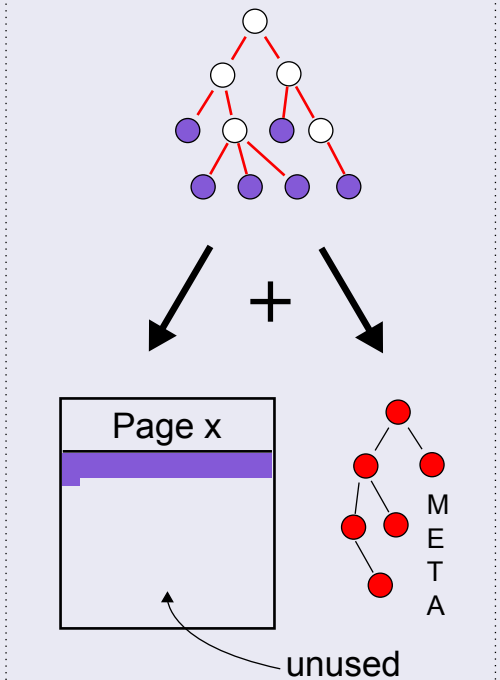
Configuration (Step 2)

Model 2 - "Elementless"

- Dedicated storage
- Indexes optional
- Path synopsis (PCRs) mandatory
- Solely leaf nodes stored
- Inner structure rebuilt on-demand

- Reduced redundancy reduces storage space
- Retain all document properties

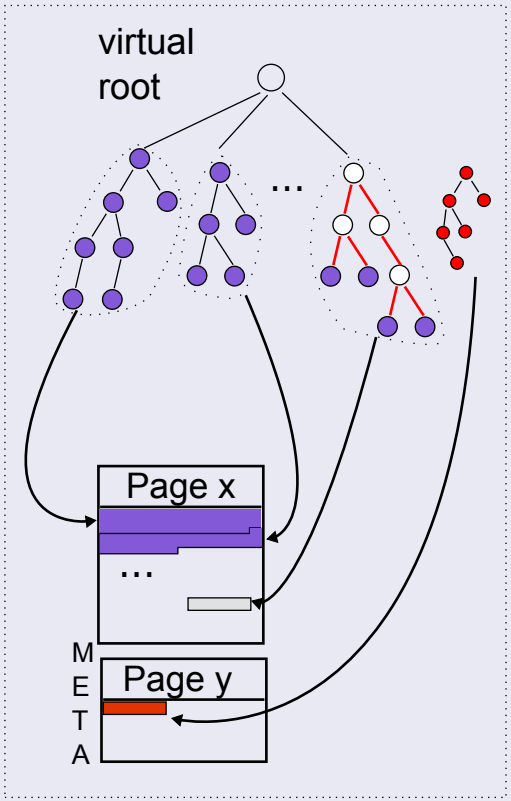
- Limited to path synopsis usage (number of path classes)



Configuration (Step 2)

Model 3 - Collections

- (small) Documents (of one domain) combined
- Virtual root node
- Combination of single and elementless documents possible
- Optional indexes for all documents

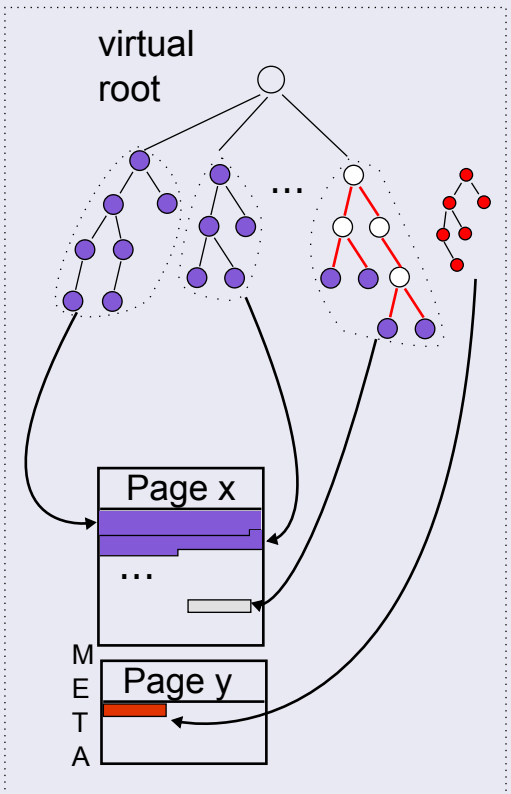


Configuration (Step 2)

Model 3 - Collections

- (small) Documents (of one domain) combined
- Virtual root node
- Combination of single and elementless documents possible
- Optional indexes for all documents

- Storage space consumption (especially collections containing many small documents)



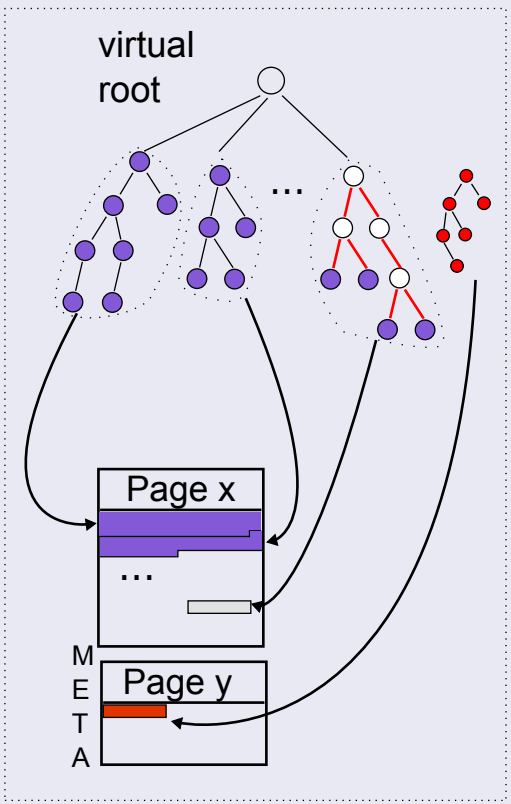
Configuration (Step 2)

Model 3 - Collections

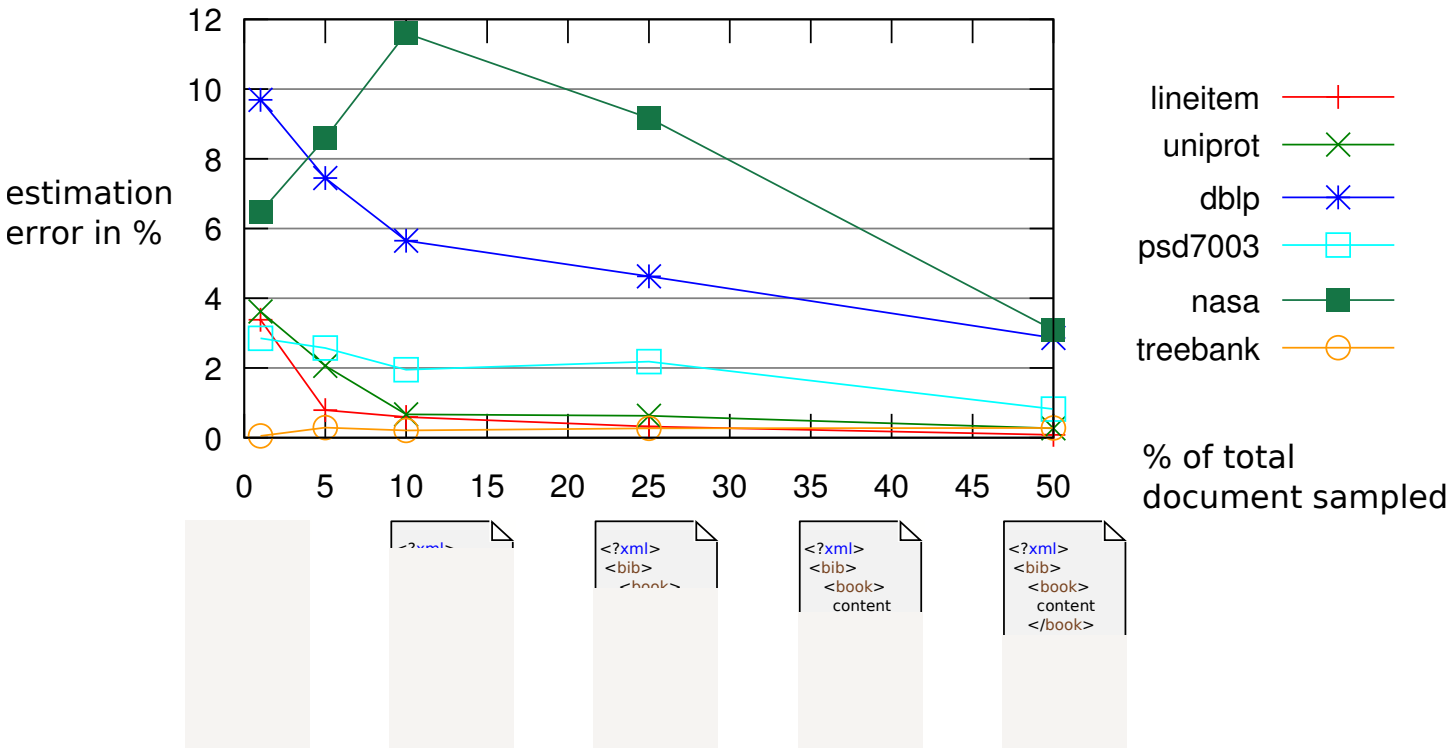
- (small) Documents (of one domain) combined
- Virtual root node
- Combination of single and elementless documents possible
- Optional indexes for all documents

- Storage space consumption (especially collections containing many small documents)

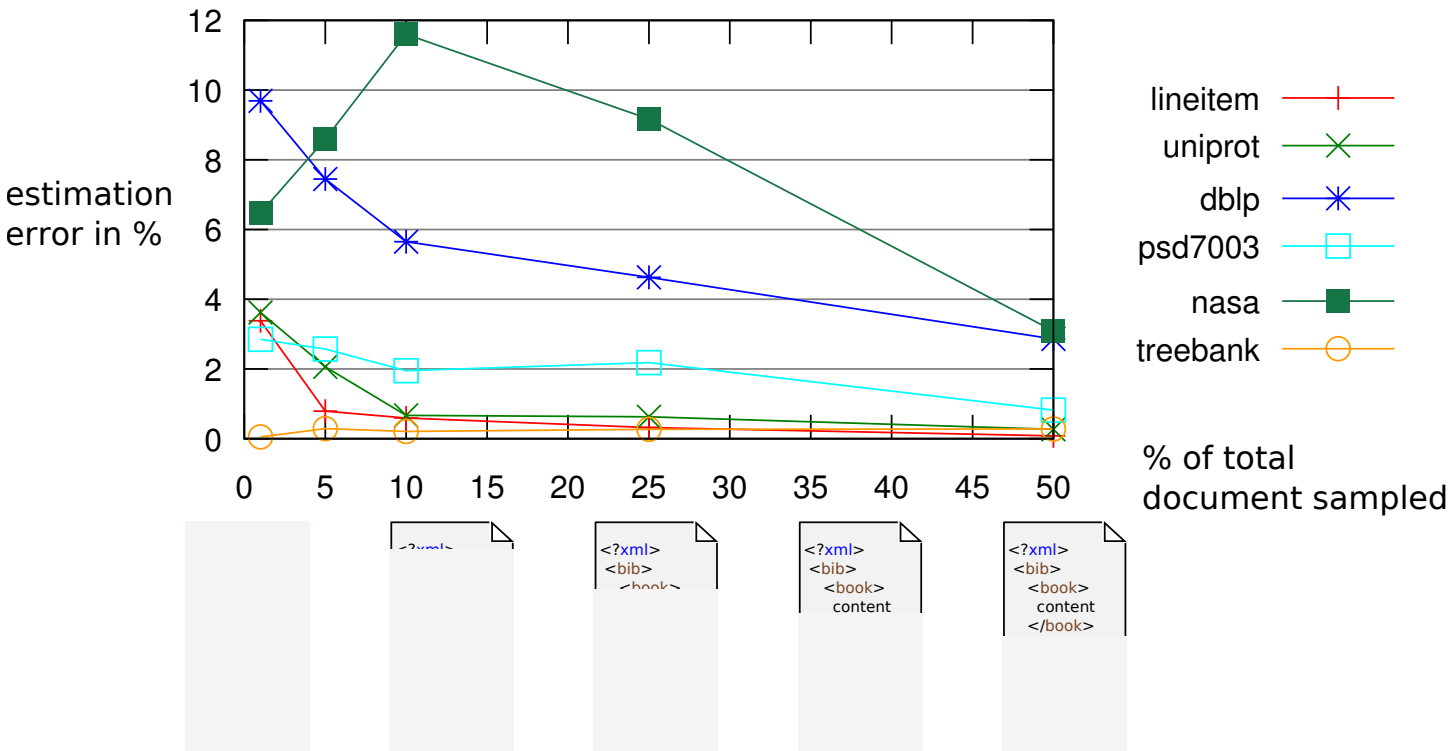
- Metadata management required
- Higher abstraction level (mostly compensated by prefix compression)



Sampling Accuracy



Sampling Accuracy

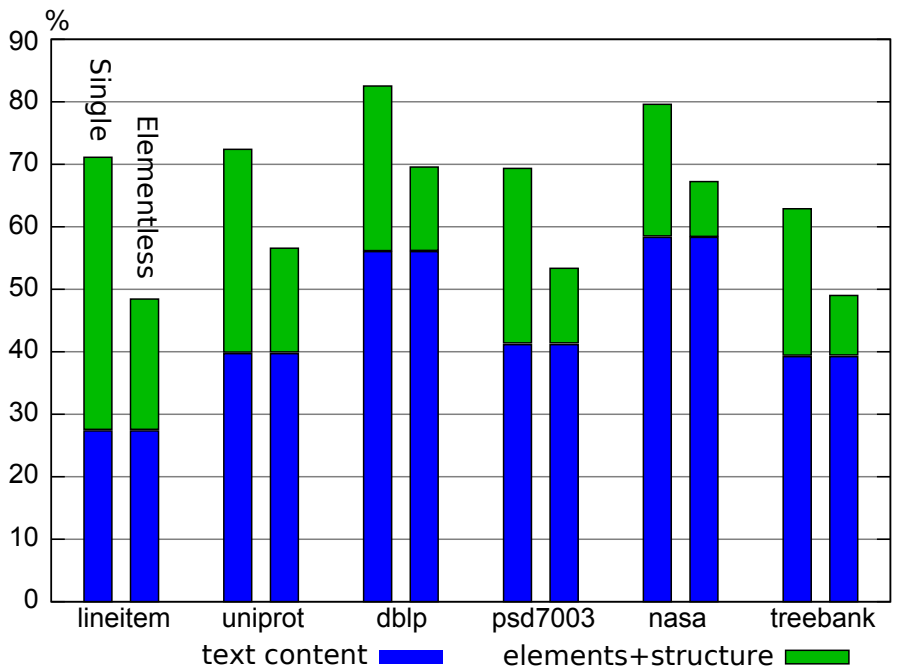


- Low sampling error on $< 10\%$ of data
- Extrapolation lead to fairly good and fast approximations

Storage Space Consumption Analysis

Single and Elementless Storage

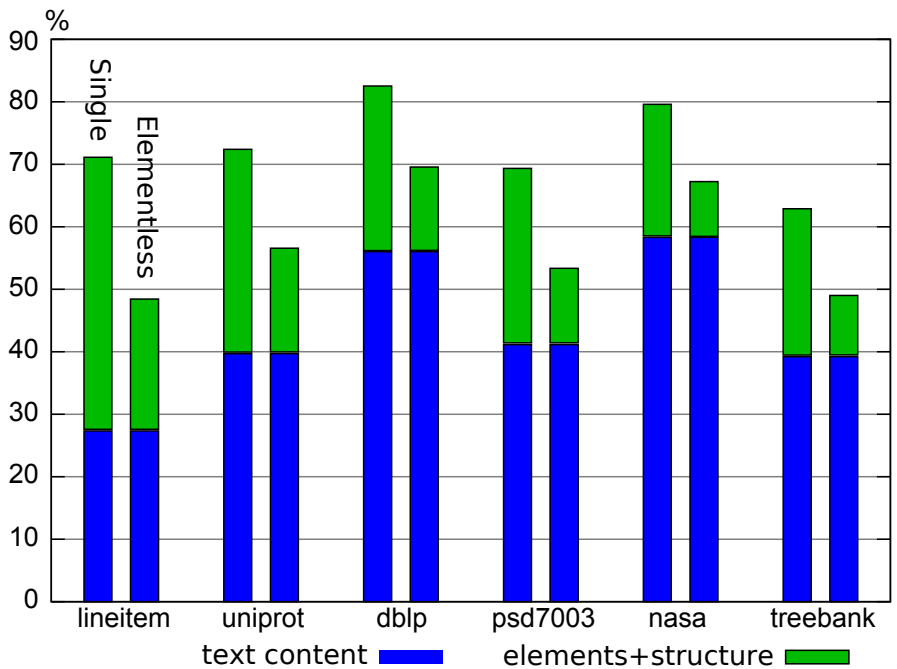
- On average Elementless saves up to 50% of structural information
- DeweyIDs are prefix compressed and element names substituted by IDs
- 100% refers to "naive" storage (full names, IDs)



Storage Space Consumption Analysis

Single and Elementless Storage

- On average Elementless saves up to 50% of structural information
- DeweyIDs are prefix compressed and element names substituted by IDs
- 100% refers to "naive" storage (full names, IDs)

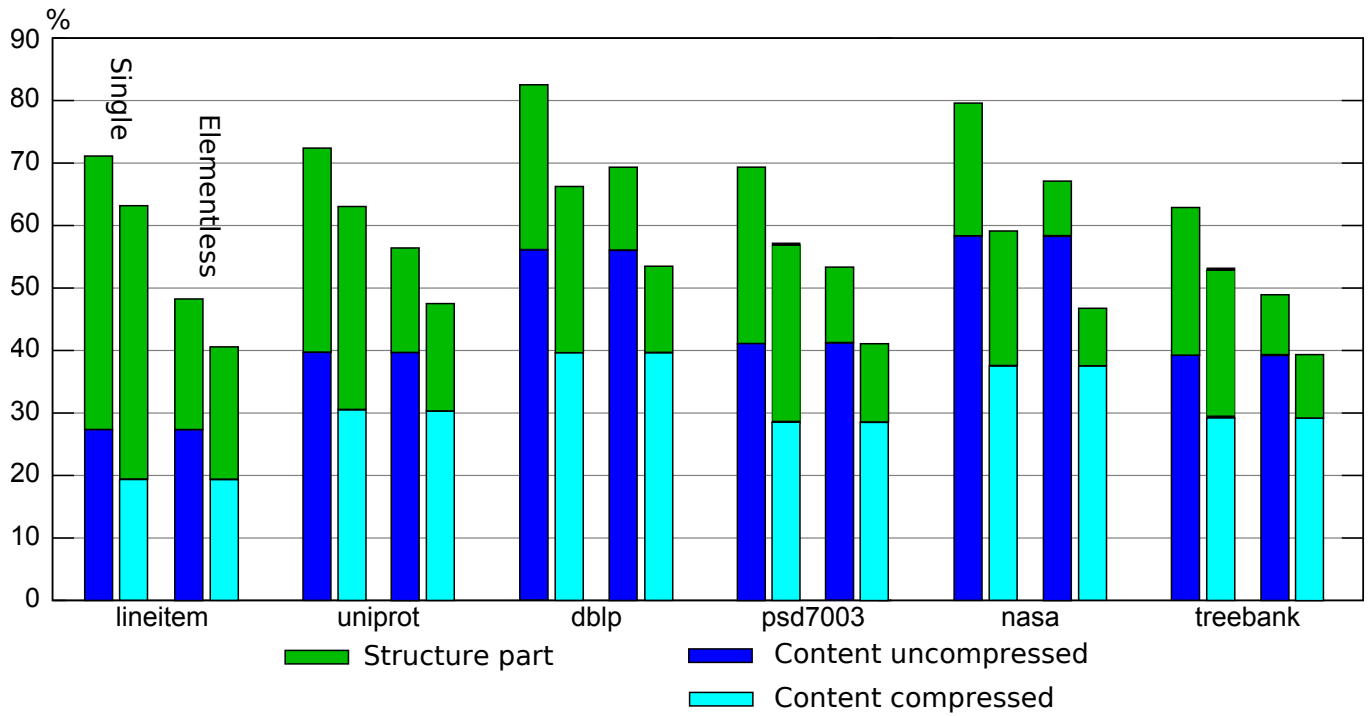


Collection Storage

Experiments on 527 randomly selected documents (wikipedia.com) reduced unused storage space up to **94.5%** by using collections! (compared to Single Document Storage)

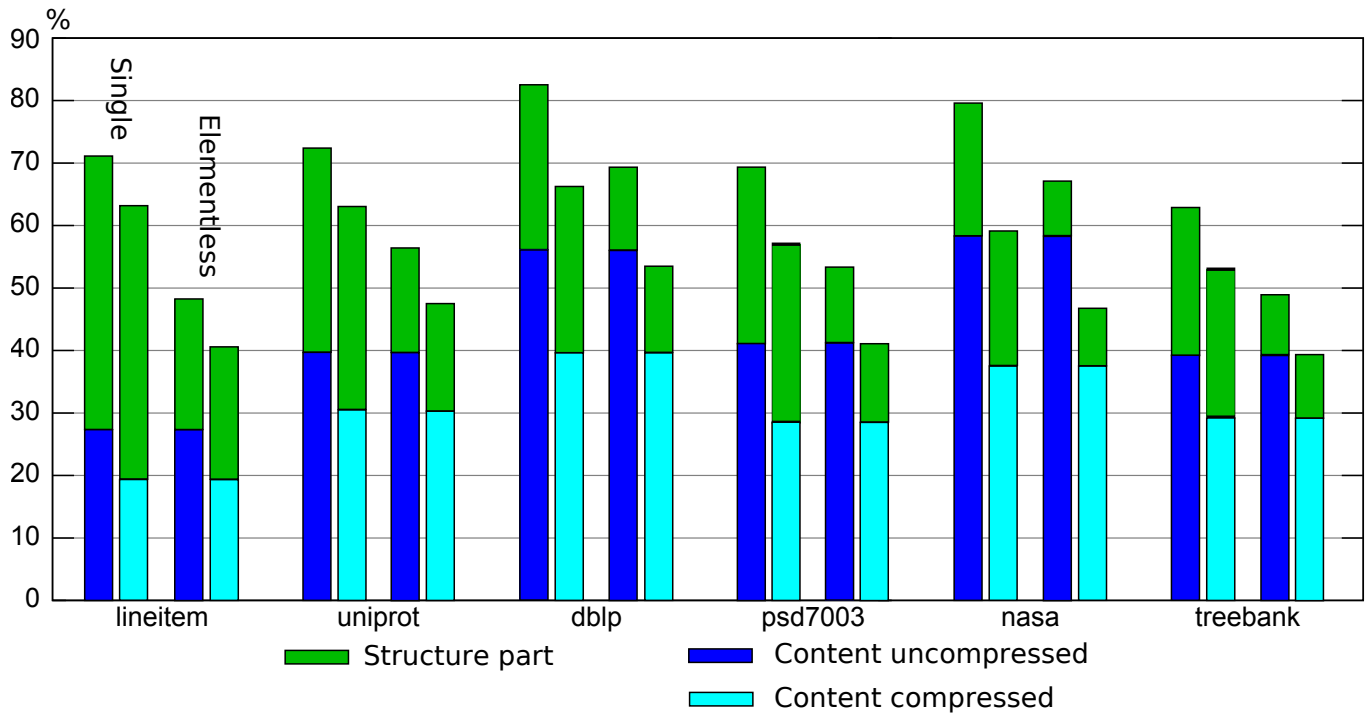
Storage Space Consumption Analysis - Compression

Character-Based Compression for Single and Elementless



Storage Space Consumption Analysis - Compression

Character-Based Compression for Single and Elementless



- Even storage and reconstruction timings could be reduced by $20 - 30\%$
- Adjusting storage parameters improves overall performance!

Thanks for your attention!

Questions?