

IEEE Focs '05
+
WWW '06

XML

Compression and Indexing

Paolo Ferragina

Dipartimento di Informatica, Università di Pisa

[Joint with F. Luccio, G. Manzini, S. Muthukrishnan]

Paolo Ferragina, Università di Pisa

Five years ago...

[now, J. ACM 05]



FOCS 2000



The 41st Annual Symposium on Foundations of Computer Science

Opportunistic Data Structures with Applications

P. Ferragina, G. Manzini

An XML excerpt

```
<dblp>
  <book>
    <author> Donald E. Knuth </author>
    <title> The TeXbook </title>
    <publisher> Addison-Wesley </publisher>
    <year> 1986 </year>
  </book>
  <article>
    <author> Donald E. Knuth </author>
    <author> Ronald W. Moore </author>
    <title> An Analysis of Alpha-Beta Pruning </title>
    <pages> 293-326 </pages>
    <year> 1975 </year>
    <volume> 6 </volume>
    <journal> Artificial Intelligence </journal>
  </article>
  ...
</dblp>
```

Paolo Ferragina, Università di Pisa

A key concern: Verbosity...

INDUSTRY TRENDS

Will Binary XML Speed Network Traffic?

David Geer

With an ability to enable data interoperability between applications on different platforms, XML has become integral to many critical enterprise technologies. For example, XML enhances e-commerce, communication between businesses, and companies' internal integration of data from multiple sources, noted analyst Randy Hillier with Forrester Research, a market-analysis firm.

XML use is thus increasing rapidly. Analyst Ron Schmelzer with market-research firm ZapThink predicted XML will increase 3 percent of global network traffic in 2003 to 24 percent by 2006, as Figure 1 shows, and to at least 40 percent by 2008.

However, XML's growing implementation raises a key concern: Because it provides considerable metadata about each element of a document's content, XML files can include a great deal of data. They can thus be inefficient to process and can burden a company's network, processor, and storage infrastructures, explained IBM Distinguished Engineer Jerry Cuomo.

"XML is commonly viewed as too much space it needs to use for the amount of data that it is sending," said Jeff Larch, chief technology officer of Leader Technologies, which uses XML in information applications.

Schmelzer, said Hillier, "XML's verbosity approved the standard's first version in 1996.

A key factor driving the standard's development was increased Internet and network usage requiring companies on different platforms to be able to communicate. Many businesses also wanted to make legacy data available to new Web-based applications.

How XML works

XML is a markup language that can define specific languages for use with structured data in related documents. An organization can develop its own XML-based language with its own set of underlying tags. For example, a group of retailers could agree to use the same set of tags for categories of data—such as "customer names" or "price per unit"—on a product order form.

A typical XML file also includes information about a document, unrelated to content, such as a description used and the properties that must be processed, or links to or as part of processing the file.

The XML document type definition (DTD), which controls and manages XML's development as a standard, and Sun Microsystems are working on binary XML formats.

Some industry observers have expressed concern that multiple formats or proprietary implementations of binary XML could lead to incompatible versions, which would reduce the value of the technology.

XML'S PROBLEMS

The W3C started work on XML in 1996 as a way to enable data interoperability over the Internet. The con-

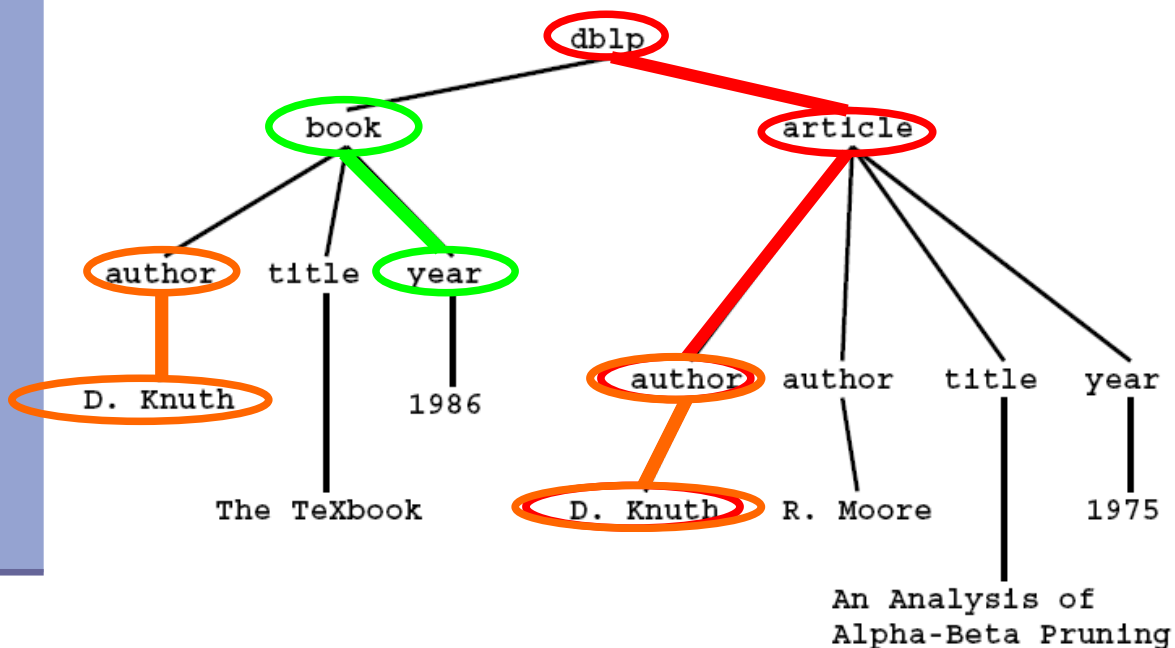
cern is that the XML document type definition describes a document's message rules—identifying messages, setting which elements can appear, and noting how they are structured—so that applications that create work with it. XML documents are written and stored as text, and documents are read via either text editors or XML parsers.

By enabling cross-platform communications, XML also makes the need to write multiple versions of documents or to use costly and complex middleware. However, the files contain considerably more information than just the content they are communicating.

XML is the basis for important technologies such as Web services and important standards such as the Simple Object Access Protocol, a way for a program to communicate with a program running in another by using HTTP and XML as the information-exchange mechanisms.

However, XML's growing implementation raises a key concern: Because it provides considerable metadata about each element of a document's content, XML files can include a great deal of data. They can thus be inefficient to process and can burden a company's network, processor, and storage infrastructures, explained IBM Distinguished Engineer Jerry Cuomo.

A tree interpretation...



- ✓ XML document exploration \equiv Tree navigation
- ✓ XML document search \equiv Labeled subpath searches

Paolo Ferragina, Università di Pisa

Subset of XPath [W3C]

The Problem

We wish to devise a **compressed representation** for a labeled tree T that efficiently supports some operations:

- ✓ **Navigational operations**: $\text{parent}(u)$, $\text{child}(u, i)$, $\text{child}(u, i, c)$
- ✓ **Subpath searches**: given a sequence Π of k labels
- ✓ **Content searches**: subpath + substring search
- ✓ **Visualization operation**: given a node, visualize its descending subtree

- ✓ **XML-aware compressors** (like XMill, XmlPom, ScmPom...)

➤ need the w

XML-native search engines

need this tool as a core block for

query optimization and (compressed) storage

- ✓ **XML-queriable**

➤ **poor comp**

- ✓ **Summary indexes** (like Dataguide, 1-index or 2-index)

➤ **large space** and **do not** support “content” searches

- ✓ **Theoretically** do exist many solutions, starting from [Jacobson, IEEE Focs '89]

➤ **no subpath/content** searches, and **poor performance** on labeled trees

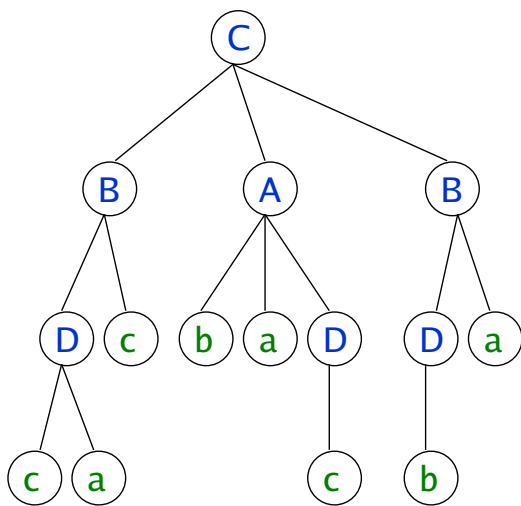
Paolo Ferragina, Università di Pisa

A transform for “labeled trees” [Ferragina et al, IEEE Focs '05]

- ✓ We proposed the **XBW-transform** that mimics on trees the nice structural properties of the Burrows-and-Wheeler Transform on strings (do you know *bzip* !?).
- ✓ The XBW **linearizes** the tree **T** in **2 arrays** s.t.:
 - ✓ the **compression** of T *reduces to* use any **k-th order entropy compressor** (*gzip, bzip,...*) over these two arrays
 - ✓ the **indexing** of T *reduces to* implement simple **rank/select** query operations over these two arrays

Paolo Ferragina, Università di Pisa

The XBW-Transform



S	S_π
ϵ	ϵ
C	C
B	BC
D	DBC
c	DBC
a	DBC
c	BC
A	C
b	AC
a	AC
D	AC
c	DAC
B	C
D	BC
b	DBC
a	BC

Permutation of tree nodes

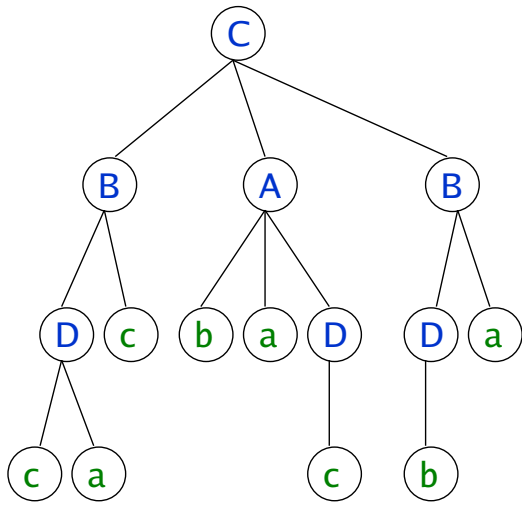
upward labeled paths

Step 1.

Visit the tree in pre-order.
For each node, write down its label and the labels on its upward path

Paolo Ferragina, Università di Pisa

The XBW-Transform



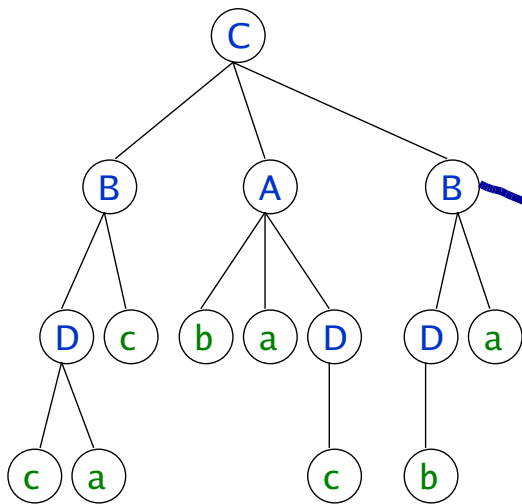
S	S_π
ϵ	ϵ
C	AC
b	AC
a	AC
D	AC
D	BC
c	BC
D	BC
a	BC
B	C
A	C
B	C
c	DAC
c	DBC
a	DBC
b	DBC

↑
upward labeled paths

Step 2.
Stably sort according to S_π

Paolo Ferragina, Università di Pisa

The XBW-Transform



S_{last}	S	S_π
1	ϵ	ϵ
0	C	AC
0	b	AC
0	a	AC
1	D	AC
0	D	BC
1	c	BC
0	D	BC
1	a	BC
0	B	C
0	A	C
1	B	C
1	c	DAC
0	c	DBC
1	a	DBC
1	b	DBC

XBW can be built and inverted in *optimal* $O(t)$ time

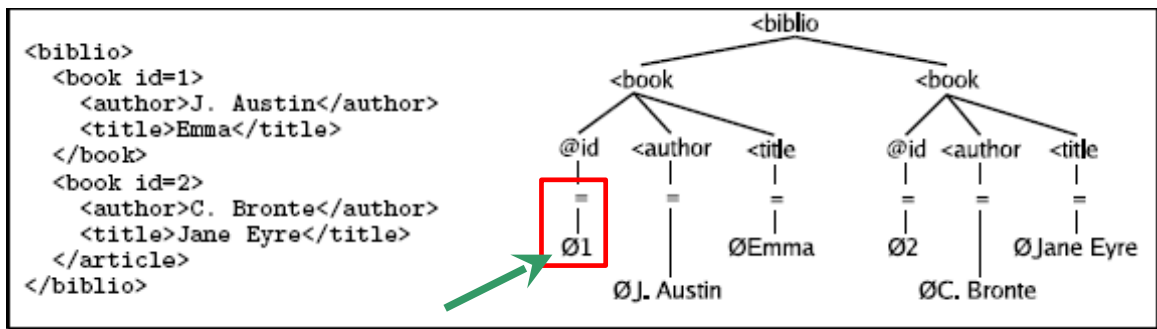
Key fact
Add a binary array S_{last} , marking the rows corresponding to *last* children

XBW

XBW takes *optimal* $t \log |\Sigma| + 2t$ bits

Paolo Ferragina, Università di Pisa

XBzip – a simple XML compressor



Rk	S_{tag}	S_{α}	S_{-}
1	1	<biblio	empty string
2	1	-	<author<book<biblio
3	1	-	<author<book<biblio
4	0	<book	<biblio
5	1	<book	<biblio
6	0	@id	<book<biblio
7	0	<author	<book<biblio
8	1	<title	<book<biblio
9	0	@id	<book<biblio
10	0	<author	<book<biblio
11	1	<title	<book<biblio
12	1	-	<title<book<biblio
13	1	-	<title<book<biblio
14	1	-	@id<book<biblio
15	1	-	@id<book<biblio
16		ØJ. Austin	-<author<book<biblio
17		ØC. Bronte	-<author<book<biblio
18		ØEmma	-<title<book<biblio
19		ØJane Eyre	-<title<book<biblio
20		Ø1	-@id<book<biblio
21		Ø2	-@id<book<biblio

Tags, Attributes and symbol =

Pcdata

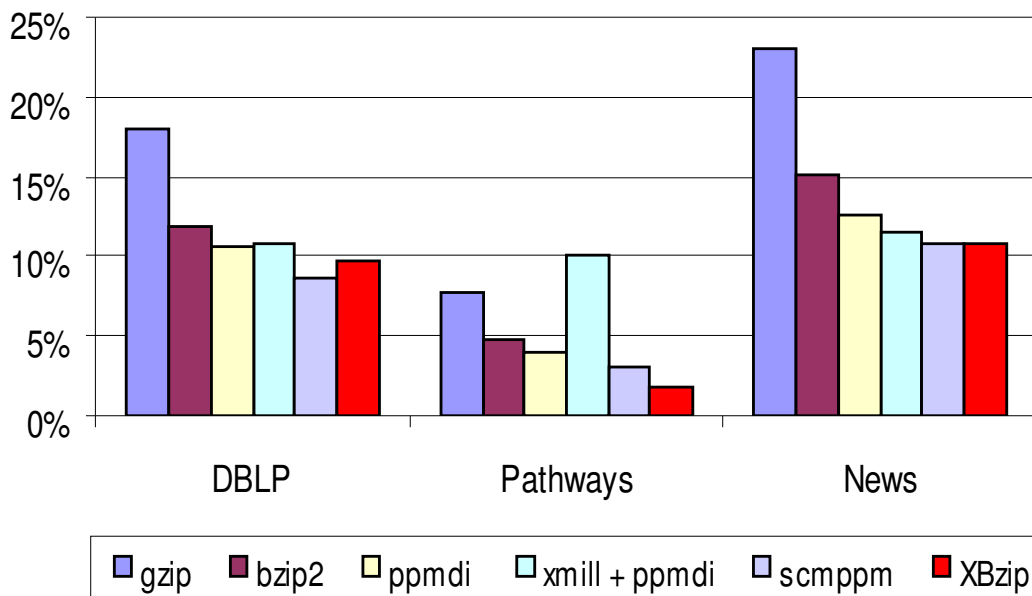
XBW is compressible:

- ① S_{α} and S_{pcdata} are locally homogeneous
- ② S_{last} has some structure

Paolo

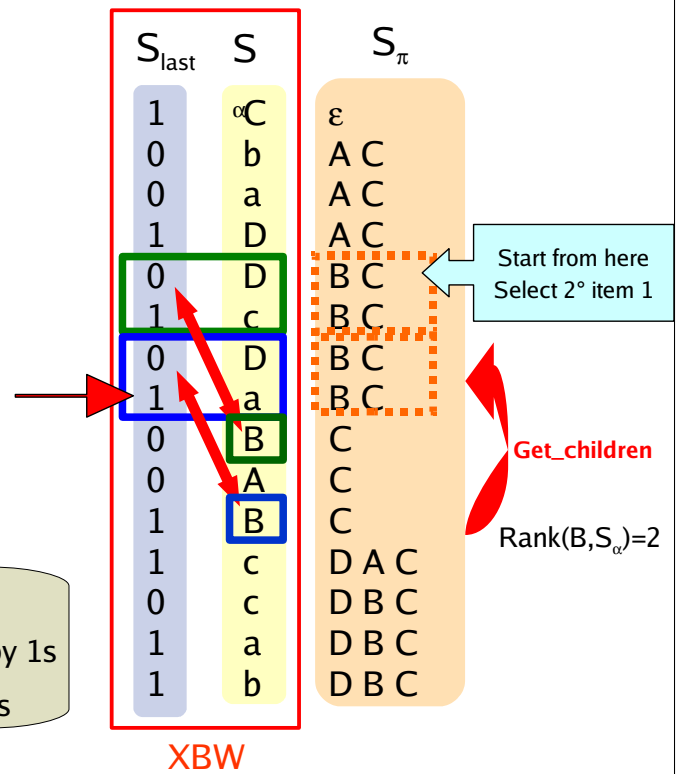
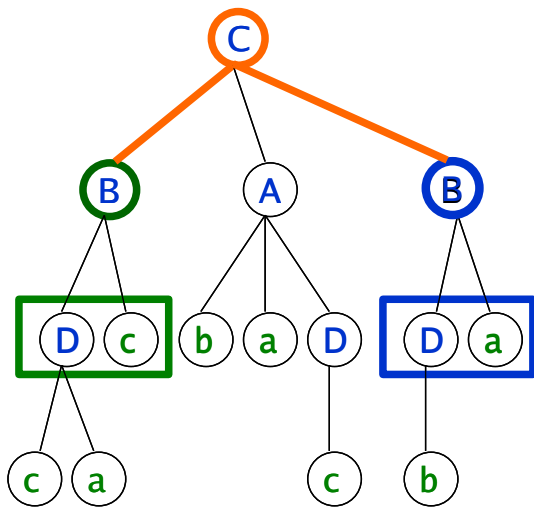
XBzip = XBW + PPMd

[Ferragina et al, WWW '06]



String compressors are not so bad: within 5%

XBW is navigational



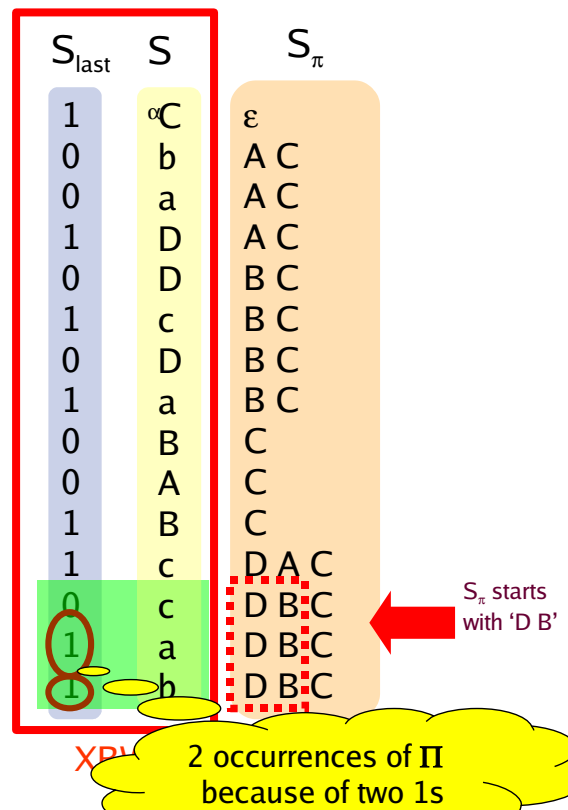
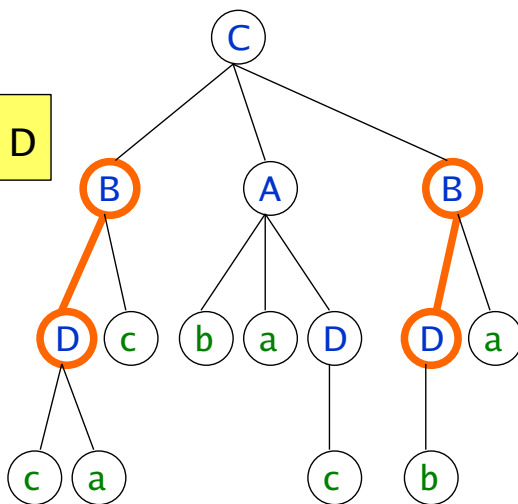
Two useful properties:

- Children are contiguous and delimited by 1s
- Children reflect the order of their parents

Paolo Ferragina, Università di Pisa

XBW is searchable

$\Pi = B D$



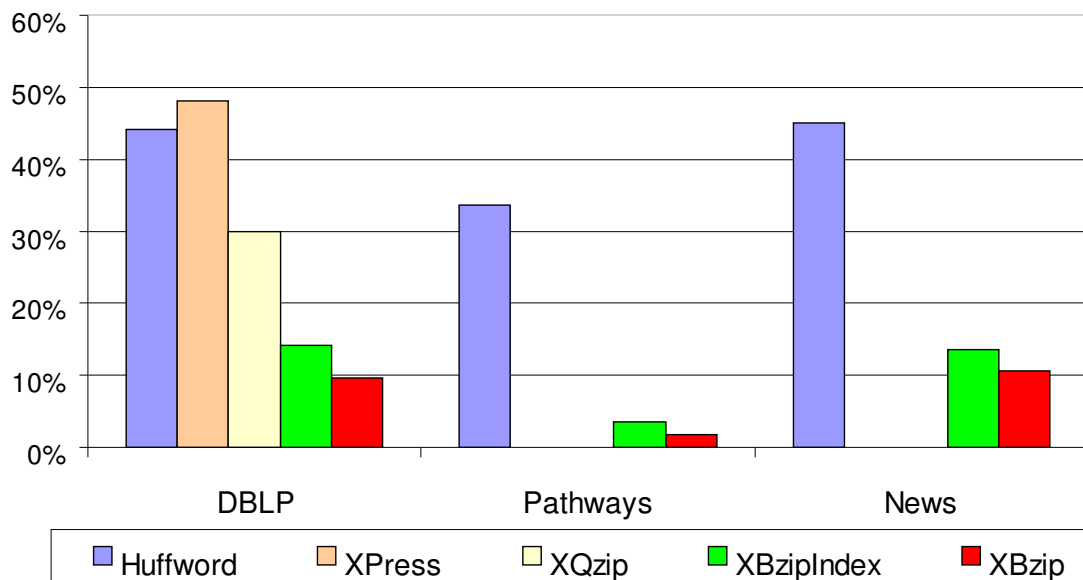
XBW indexing [reduction to string indexing]:

- Store succinct and efficient Rank and Select data structures over these three arrays

Paolo Ferragina, Università di Pisa

XBzipIndex: XBW + FM-index

[Ferragina *et al*, WWW '06]



DBLP: 1.75 bytes/node, Pathways: 0.31 bytes/node, News: 3.91 bytes/node

Paolo Ferragina, Università di Pisa

Upto 36% improvement in compression ratio
Query (counting) time \cong 8 ms, Navigation time \cong 3 ms

The overall picture on Compressed Indexing...

Data type

Text

Labeled Tree

This is a powerful paradigm to design compressed indexes:

2. Transform the input in few arrays (via BWT or XBW)
3. Index (+ Compress) the arrays to support rank/select ops

Paolo Ferragina, Università di Pisa

Theory: Soda '06 (2), Cpm '06 (2), Icalp '06 (2), DCC '06 (1)
Experimental: Wea '06 (2)