RuSSIR 2010

XML Retrieval

Mounia Lalmas
mounia@acm.org

Outline

- Introduction to XML, basics and standards
- Document-oriented XML retrieval
- Evaluating XML retrieval effectiveness
- Going beyond XML retrieval

Outline

- **Introduction to XML, basics and standards**
- Document-oriented XML retrieval
- Evaluating XML retrieval effectiveness
- Going beyond XML retrieval

Introduction to XML, basics and standards

- What is XML?
- Database vs. information retrieval
- Document Type Definition
- XML Schema
- Querying XML Data

XML (eXtensible Markup Language)

- A **meta-language** (a language for describing other languages)
  XML is able to represent a mix of structured and text (unstructured) information
- Defined by the WWW Consortium (**W3C**)
  developed by a W3C working group, headed by James Clark.
- XML 1.0 became a W3C Recommendation on February 10, 1998
- At present XML is the *de facto* standard markup language.

XML: eXtensible Mark-up Language

- Meta-language (user-defined tags) currently being adopted as the document format language by W3C
- Used to describe **content and structure** (and not layout)
- Grammar described in DTD (→ used for validation)

```
<lecture>
   <title> Structured Document Retrieval </title>
   <author> <fnm> Smith </fnm> <snm> John </snm> </author>
   <chapter>
      <title> Introduction into SDR </title>
      <paragraph> …. </paragraph>
      …
   </chapter> …
</lecture>
```

```
<!ELEMENT lecture (title, author+,chapter+)
>
<!ELEMENT author (fnm*,snm)>
<!ELEMENT fnm #PCDATA>
…
```

XML: eXtensible Mark-up Language

- Use of XPath notation to refer to the XML structure

chapter/title: title is a direct sub-component of chapter
//title: any title
chapter//title: title is a direct or indirect sub-component of chapter
chapter/paragraph[2]: any direct second paragraph of any chapter
chapter/*: all direct sub-components of a chapter

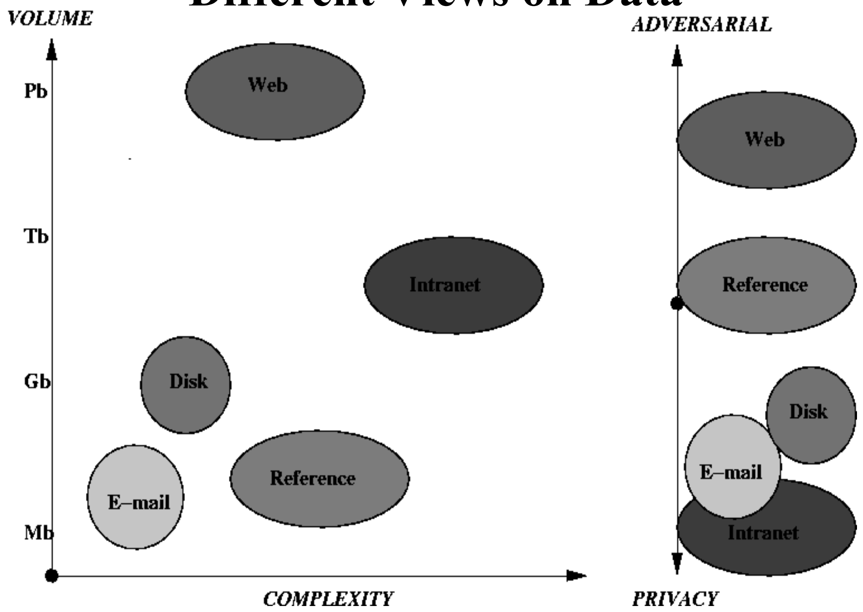
```
<lecture>
    <title> Structured Document Retrieval </title>
    <author> <fnm> Smith </fnm> <snm> John </snm> </author>
    <chapter>
        <title> Introduction into SDR </title>
        <paragraph> …. </paragraph>
        …
    </chapter> …
</lecture>
```

XML

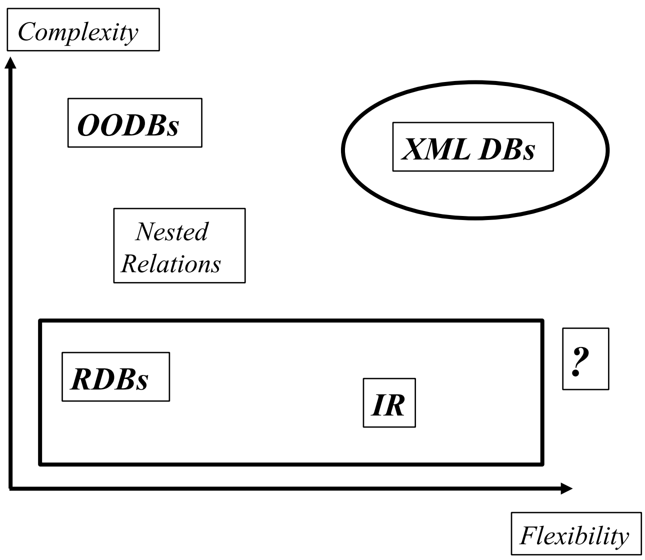
- XML applications: *data interchange, digital libraries, content management, complex documentation, etc.*
- XML repositories: *Library of Congress collection, SIGMOD DBLP, IEEE INEX collection, LexisNexis, …*

(http://www.w3.org/XML/)

Different Views on Data



Data and Databases



DB vs. IR

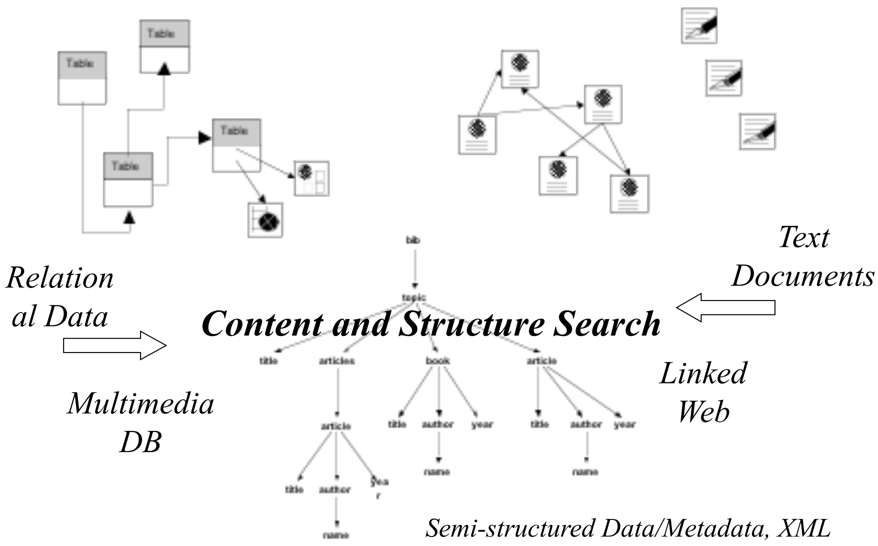
- DBs allow structured querying
- Queries and results (tuples) are different objects
- Soundness & completeness expected
- All results are equally good
- User is expected to know the structure (Enterprise)

- IR only supports unstructured querying
- Queries and results are both documents
- Results are usually imprecise & incomplete
- Some results are more relevant than others
- User is expected to be dumb (Web)

The Notion of Relevance

- Data retrieval: semantics tied to syntax
- Information retrieval: ambiguous semantics
- Relevance:
  - Depends on the user
  - Depends on the context (task, time, etc)
  - Corollary: The Perfect IR System does not exist

Convergent Path?



Problems of the IR view

- Very simple query language
  - Is natural language the solution?
- No query optimization
- Does not handle the complete answer
- No types

Problems of the DB view

- The syndrome of the formal model
  - Model is possible because of structure
- The syndrome of "search then rank"
  - Large answers
  - Optimization is useless
  - Quality vs. Speed
  - E.g. XQuery
- What is a Database?
- Are RDBs really a special case of IR systems?
  - Full text over fields

DB and IR view

- Data-centric view
  - XML as exchange format for structured data
  - Used for messaging between enterprise applications
  - Mainly a recasting of relational data
- Document-centric view
  - XML as format for representing the logical structure of documents
  - Rich in text
  - Demands good integration of text retrieval functionality
- Now increasingly both views (DB+IR)

Possible Architectures

- IR on top of RDBs
- IR supported via functions in an RDB
- IR on top of a relational *storage* engine

- Middleware layer on top of RDB & IR systems
- RDB functionality on top of an IR system
- Integration via an XML database & query language

Data-Centric XML Documents: Example

```
<CLASS name="DCS317" num_of_std="100">
    <LECTURER lecid="111">Thomas</LECTURER>
    <STUDENT marks="70" origin="Oversea">
          <NAME>Mounia</NAME>
     </STUDENT>
    <STUDENT marks="30" origin="EU">
          <NAME>Tony</NAME>
    </STUDENT>
</CLASS>
```

Document-Centric XML Documents: Example

```
<CLASS name="DCS317" num_of_std="100">
    <LECTURER lecid="111">Mounia</LECTURER>
    <STUDENT studid="007" >
        <NAME>James Bond</NAME> is the best student in the
        class. He scored <INTERM>95</INTERM> points out of
        <MAX>100</MAX>. His presentation of <ARTICLE>Using
        Materialized Views in Data Warehouse</ARTICLE> was
        brilliant.
    </STUDENT>
    <STUDENT stuid="131">
        <NAME>Donald Duck</NAME> is not a very good
        student. He scored <INTERM>20</INTERM> points…
        </STUDENT>
</CLASS>
```

Document-centric XML retrieval

- Documents marked up as XML
  - E.g., assembly manuals, journal issues …
- Queries are user information needs
  - E.g., give me the Section (element) of the document that tells me how to change a brake light
- Different from well-structured XML queries where one tightly specifies what he/she is looking for.

XML World

- DTD: Document Type Definition
- XSchema: Data Schema
- DOM: Document Object Model
- SOX: Schema for Object-oriented XML
- Others: XPointer, XSL, ….

- XLST: to transform XML
- XPath: to extract XML elements and content
- XQuery: to query XML

XML

- Documents have tags giving extra information about sections of the document
  <title> XML </title> <slide> Introduction …</slide>

- Derived from SGML (Standard Generalized Markup Language) but simpler to use

- Extensible, unlike HTML
  users can add new tags, and *separately* specify how the tag should be handled for display

- Goal was (is?) to replace HTML as the language for publishing documents on the Web

XML

- The ability to specify new tags, and to create **nested tag structures** made XML a great way to exchange data, not just documents.
  - many of the use of XML has been in data exchange applications, and not just a replacement for HTML
- Tags make data **self-documenting**

Example of an XML document

(from database)

```
<?xml version= “1.0” encoding= “UTF-8” standalone= “yes”?>
<?xml:stylesheet type = “text/xsl” href = “staff_list.xsl”?>
<!DOCTYPE STAFFLIST SYSTEM “staff_list.dtd”>
<STAFFLIST>
    <STAFF branchNo = “B005”>
            <STAFFNO>SL21</STAFFNO>
                <NAME>
                    <FNAME>John</FNAME><LNAME>White</LNAME>
                </NAME>
            <POSITION>Manager</POSITION>
            <DOB>1-Oct-45</DOB>
            <SALARY>30000</SALARY>
    </STAFF>
    <STAFF branchNo = “B003”>
            <STAFFNO>SG37</STAFFNO>
            <NAME>
                <FNAME>Ann</FNAME><LNAME>Beech</LNAME>
            </NAME>
            <POSITION>Assistant</POSITION>
            <SALARY>12000</SALARY>
    </STAFF>
</STAFFLIST>
```

XML - Elements

- **Tag**: label for a section of data
- **Element**: section of data beginning with <tagname> and ending with matching </tagname>

- Elements must be properly **nested**
  - Proper nesting
    - <account> … <balance> …. </balance> </account>
  - Improper nesting
    - <account> … <balance> …. </account> </balance>
  - Formally: every start tag must have a unique matching end tag that is in the context of the same parent element.

- Every document must have a single top-level element

Example of Nested Elements

```
<bank>
   <customer>
      <customer-name> Monz </customer-name>
      <customer-street> Mile End </customer-street>
      <customer-city>   London </customer-city>
      <account>
         <account-number> A-102 </account-number>
         <branch-name>     QMUL </branch-name>
         <balance>         400 </balance>
      </account>
      <account>
         …
      </account>
   </customer>
      .
      .
</bank>
```

XML - Elements

- **Mixture of text with sub-elements:**

```
<account>
   This account is seldom used any more.
    <account-number> A-102</account-number>
    <branch-name> QMUL</branch-name>
    <balance>400 </balance>
</account>
```

  - **Useful for document markup but discouraged for data representation**

XML - Attributes

- **Elements can have attributes**

```
<account acct-type = “checking” >
      <account-number> A-102 </account-number>
      <branch-name>QMUL </branch-name>
      <balance> 400 </balance>
  </account>
```

- **Attributes are specified by** name=value **pairs inside the starting tag of an element**

- **An element may have several attributes, but each attribute name can only occur once**

```
<account  acct-type = “checking”  monthly-fee=“5”>
```

XML - Attributes Vs. Elements

- In the context of **documents**, attributes are part of markup, while element contents are part of the basic document contents
- In the context of **data representation**, the difference is unclear and may be confusing
  - <account account-number = “A-101”> …. </account>
  - <account>
<account-number>A-101</account-number> …
</account>
- Suggestion: use attributes for identifiers of elements, and use elements for contents

XML – Other Syntax

- Elements **without sub-elements or text content** can be abbreviated by ending the start tag with a /> and deleting the end tag

  <account number=“A-101” branch=“QMUL” balance=“200 />

- **Comments**: enclosed in <!– and --> tags.
- **CDATA sections**: instructs XML processor to ignore markup characters and pass enclosed text directly to application.

  <![CDATA[<account> … </account>]]>

XML – Ordering

- In XML, elements are ordered.
- In contrast, in XML attributes are unordered.

Document Type Definition (DTD)

- **Type of an XML document can be specified using a DTD**
- **DTD constraints structure of XML data**
  - What elements can occur?
  - What attributes can/must an element have?
  - What sub-elements can/must occur inside each element, and how many times?
- **DTD does not constrain data types**
  - All values represented as strings in XML
- **DTD syntax**

  <!ELEMENT element-name (subelements-specification) >

  <!ATTLIST element-name (attributes) >

Element Specification in DTD

- **Sub-elements can be specified as**
  - names of elements
  - #PCDATA (parsed character data), i.e., character strings
  - EMPTY (no sub-elements) or ANY (anything can be a sub-element)
- **Example**

  <! ELEMENT depositor (customer-name account-number)>
  <! ELEMENT customer-name (#PCDATA)>
  <! ELEMENT account-number (#PCDATA)>
- **Sub-element specification may have regular expressions**

  <!ELEMENT bank ((account | customer | depositor)+)>

  "|" - alternatives
  "+" - 1 or more occurrences
  "*" - 0 or more occurrences
  "?" - 0 or 1 occurrence

Attribute Specification in DTD

- For each attribute
  - Name
  - Type of attribute
    - CDATA
    - ID (identifier) or IDREF (ID reference) or IDREFS (multiple IDREFs)
  - Whether
    - mandatory (#REQUIRED)
    - has a default value (value),
    - or neither (#IMPLIED)
- Examples

  <!ATTLIST account acct-type CDATA "checking">
  <!ATTLIST customer
  customer-id ID # REQUIRED
  accounts IDREFS # REQUIRED >

DTD Example

mail.dtd

```
<!ELEMENT message
   (urgent?, subject,
    body)>
<!ELEMENT subject
   (#PCDATA)>
<!ELEMENT body
   (ref|#PCDATA)*>
<!ELEMENT ref
   (#PCDATA)>
<!ELEMENT urgent
   EMPTY>
<!ATTLIST message
 date DATE #IMPLIED
 sender CDATA #REQUIRED
 receiver CDATA #REQUIRED
 mtype (TXT|MM) ``TXT''>
```

Non-XML Language

Elements

Structure
- Sequence
- Nesting

Attributes

Namespaces

- XML data has to be exchanged between organizations
- Same tag name may have different meaning in different organizations, causing confusion on exchanged documents
- Specifying a unique string as an element name avoids confusion
- Better solution: use unique-name:element-name
- Avoid using long unique names all over document by using XML Namespaces

```
<bank Xmlns:FB='http://www.FirstBank.com'>
   …
   <FB:branch>
          <FB:branchname>Downtown</FB:branchname>
      <FB:branchcity>   Brooklyn   </FB:branchcity>
   </FB:branch>
   …
</bank>
```

XML Schema

- Database schemas constrain what information can be stored, and the data types of stored values
- XML documents are not required to have an associated schema
- However, schemas are very important for XML data exchange
  - otherwise, a site cannot automatically interpret data received from another site
- Two mechanisms for specifying schema language
  - Document Type Definition (DTD)
    - Widely used
  - XML Schema
    - Newer, increasing use

XML Schema

- XML Schema is a more sophisticated schema language which addresses the drawbacks of DTDs.
  - Typing of values
    - E.g. integer, string, etc
    - Also, constraints on min/max values
  - User defined types
  - Is itself specified in XML syntax, unlike DTDs
  - Is integrated with namespaces
  - Many more features
    - List types, uniqueness and foreign key constraints, inheritance ..
- BUT: significantly more complicated than DTDs, not yet as widely used.

XML Schema -Example
(from database)

```
<xsd:schema xmlns:xsd=http://www.w3.org/2001/XMLSchema>
<xsd:element name="bank" type="BankType"/>
<xsd:element name="account">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="account-number" type="xsd:string"/>
            <xsd:element name="branch-name"    type="xsd:string"/>
            <xsd:element name="balance"        type="xsd:decimal"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
….. definitions of customer and depositor ….
<xsd:complexType name="BankType">
    <xsd:sequence>
        <xsd:element ref="account"   minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="customer"  minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="depositor" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:complexType>
</xsd:schema>
```

XML Schema Example

```
<?xml version="1.0"?>
<purchaseOrder orderDate="2006-06-20">
   <shipTo country="US">
      <name>Alice Smith</name>
      <street>123 Maple Street</street>
      <city>Mill Valley</city>
      <state>CA</state>
      <zip>90952</zip>
   </shipTo>
   <billTo country="US">
      <name>Robert Smith</name>
      <street>8 Oak Avenue</street>
      <city>Old Town</city>
      <state>PA</state>
      <zip>95819</zip>
   </bill>
   <comment>Hurry, my lawn is
going wild!</comment>
```

```
  <items>
    <item partNum="872-AA">
      <productName>Lawnmower
        </productName>
      <quantity>1</quantity>
      <USPrice>148.95</USPrice>
      <comment>Confirm this is
       electric</comment>
    </item>
    <item partNum="926-AA">
      <productName>
       Baby Monitor
       </productName>
      <quantity>1</quantity>
      <USPrice>39.98</USPrice>
      <shipDate>2006-08-21</shipDate>
    </item>
  </items>
</purchaseOrder>
```

Querying and Transforming XML Data

- **Translation of information from one XML schema to another**
- **Querying on XML data**
- Standard XML querying/translation languages
  - XSLT
    - Simple language designed for translation from XML to XML and XML to HTML
  - XPath
    - Simple language consisting of path expressions
  - XQuery
    - An XML query language with a rich set of features
- Wide variety of other languages have been proposed, and some served as basis for the XQuery standard (XML-QL, Quilt, XQL, …)

Tree Model of XML Data

- Query and transformation languages based on **tree model** of XML data
- An XML document is modeled as a tree, with nodes corresponding to elements and attributes
  - Element nodes have children nodes, which can be attributes or sub-elements
  - Text in an element is modeled as a text node child of the element
  - Children of a node are ordered according to their order in the XML document
  - Element and attribute nodes (except root node) have a single parent, which is an element node
  - Root node has single child = root element of the document
- Terminology: node, children, parent, sibling, ancestor, descendant.

XPath

- XPath used to select document parts using path expressions
- Path expression = sequence of steps separated by “/”
- Result of path expression: set of values that along with their containing elements/attributes match the specified path
- Examples
  - /bank/customer/customer-name
    <customer-name>Joe</customer-name>
    <customer-name>Mary</customer-name>
  - bank/customer/customer-name/text( )
    returns the same names, but without the enclosing tags

XPath - Examples

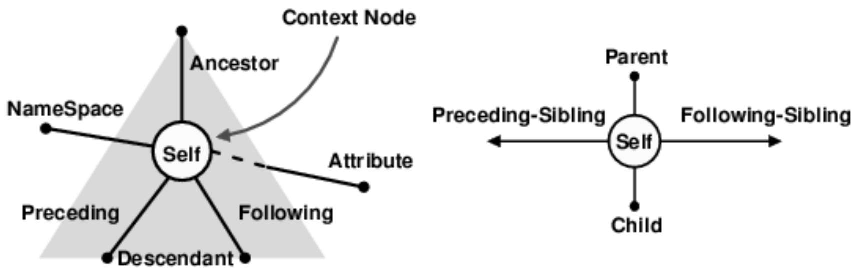
- /bank/account[balance > 400]
  returns account elements with a balance value greater than 400
- /bank/account[balance]
  returns account elements containing a balance sub-element
- /bank/account[balance > 400]/@account-number
  returns the account numbers of those accounts with balance > 400
- /bank/account[customer/count() > 2]
  returns accounts with > 2 customers

XPath

Table 30.2 Some examples of location paths.

Location path	Meaning
.	Selects the context node
..	Selects the parent of the context node
/	Selects the root node, or a separator between steps in a path
//	Selects descendants of the current node
/child::STAFF (or just /STAFF)	Selects all the STAFF elements that are children of the root
child::STAFF (or just STAFF)	Selects the STAFF element children of the context node
attribute::branchNo (or just @branchNo)	Selects the branchNo attribute of the context node
attribute::* (or just @*)	Selects all the attributes of the context node
child::STAFF[3]	Selects the third STAFF element that is a child of the context node
/child::STAFF[@branchNo = "B005"]	Selects all the STAFF elements that have an attribute with a branchNo value of B005
/child::STAFF[@branchNo = "B005"] [position()=1]	Selects first STAFF element that has an attribute with a branchNo value of B005

XML Axes



Path Expressions

```
<bib>
  <book year="1994">
    <title>TCP/IP
    <author>
      <last>Steve
      <first>W.</
    </author>
    <publisher>A
    <price> 65.9
  </book>
```

{-- XQuery uses the abbreviated syntax of XPath for path expressions --}

```
document("bib.xml")

/bib/book/author

/bib/book//*

//author[last="Stevens" and first="W."]

document("bib.xml")//author
```

XQuery

- General purpose query language for XML data
- Currently being standardized by World Wide Web Consortium (**W3C**)
- Derived from the Quilt query language, itself based on features from XPath, XML-QL, SQL, OQL, Lorel, XQL, and YATL.

SQL vs. XQuery

"Find item numbers of books"

SQL:

```
SELECT itemno
FROM items AS i
WHERE description LIKE 'Book'
ORDER BY itemno;
```

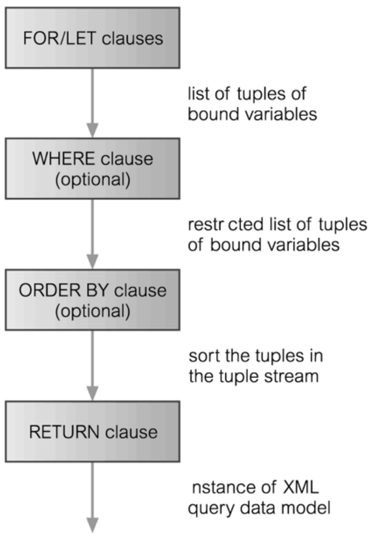
XQuery:

```
FOR $i IN //item_tuple
WHERE contains($i/description, "Books")
RETURN $i/itemno ORDERBY(.)
```

XQuery – FLWOR Expressions

- **FLWOR** (**"flower"**) **expression is constructed from**
  - **FOR**,
  - **LET**,
  - **WHERE**,
  - **ORDER BY**,
  - **RETURN**

  clauses.



Example - FLWOR Expressions

List staff at branch B005 with salary > £15,000.

```
FOR $S IN doc("staff_list.xml")//STAFF
 WHERE $S/SALARY > 15000 AND
                  $S/@branchNo = "B005"
 RETURN $S/STAFFNO
```

Example - FLWOR Expressions

List all staff in descending order of staff number.

```
FOR $S IN doc("staff_list.xml")//STAFF
ORDER BY $S/STAFFNO DESCENDING
RETURN $S/STAFFNO
```

Example - FLWOR Expressions

List each branch office and average salary at branch.

```
FOR $B IN distinct-values(doc("staff_list.xml")//@branchNo))
LET $avgSalary :=
  avg(doc("staff_list.xml")//STAFF[@branchNo = $B]/SALARY)
RETURN
    <BRANCH>
        <BRANCHNO>{ $B/text() }</BRANCHNO>,
        <AVGSALARY>$avgSalary</AVGSALARY>
    </BRANCH>
```

Example - FLWOR Expressions

List branches that have more than 20 staff.

```
<LARGEBRANCHES>
    FOR $B IN
      distinct-values(doc("staff_list.xml")//@branchNo)
      LET $S:= doc("staff_list.xml")//STAFF/[@branchNo = $B]
      WHERE count($S) > 20
      RETURN
        <BRANCHNO>{ $B/text() }</BRANCHNO>
</LARGEBRANCHES>
```

Example – Joining Two Documents

List staff along with details of their next of kin.

```
FOR  $S IN doc("staff_list.xml")//STAFF,
  $NOK IN doc("nok.xml")//NOK
WHERE $S/STAFFNO = $NOK/STAFFNO
RETURN
        <STAFFNO>{ $S, $NOK/NAME }</STAFFNO>
```

Example – Joining Two Documents

List all staff along with details of their next of kin.

```
FOR  $S IN doc("staff_list.xml")//STAFF
RETURN
   <STAFFNOK>
  { $S }
  FOR $NOK IN doc("nok.xml")//NOK
  WHERE $S/STAFFNO = $NOK/STAFFNO
  RETURN $NOK/NAME
   </STAFFNOK>
```

Why XQuery?

- Expressive power
- Easy to learn (?)
- Easy to implement (?)
- Optimizable in many environments
- Related to concepts that people already know
- Several current implementations
- The accepted W3C XML Query Language

Recap

- Components of the XML World
- Virtues and setbacks of XML Query Language
  - Powerful query language
  - But, too complex for many applications
  - Many implementations
  - Future: XQuery core?
- Any formal background?
  - Structured text models

XQuery Implementations

- Software AG's Tamino XML Query
- Microsoft, Oracle,
- Lucent Galax
- GMD-IPSI
- X-Hive
- XML Global
- SourceForge XQuench, Saxon, eXist, XQuery Lite
- Fatdog
- Qexo (GNU Kawa) - compiles to Java byte code
- Openlink, CL-XML (Common Lisp), Kweelt,...
- Soda3, DB4XML and about 15 more

Storing XML documents in databases

- Data centric and document centric XML documents
- Different ways to store XML documents
  - Flat files
  - BLOBs
  - Relational databases
  - Object-Relational databases
  - Native XML databases

http://www.rpbourret.com/xml/XMLAndDatabases.htm

Outline

- Introduction to XML, basics and standards
- **Document-oriented XML retrieval**
- Evaluating XML retrieval effectiveness
- Going beyond XML retrieval

Document-oriented XML retrieval

- Document vs. data- centric XML retrieval (recall)
- Focused retrieval
- Structured documents
- Structured document (text) retrieval
- XML query languages
- XML element retrieval
- (A bit about) user aspects

Data-Centric and Document-Centric XML

- Data with partial structure is called **semi-structured**
- XML documents are considered to be **semi-structured**

- XML documents classified as:
  - **Data centric**
  - **Document centric**

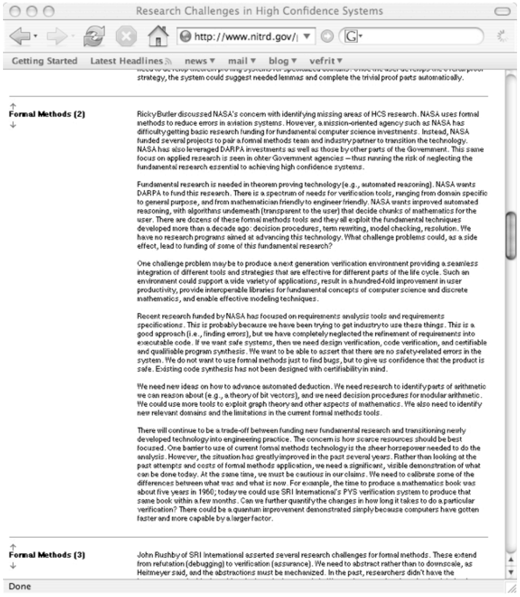
- Nowadays border between data and document centric XML documents is not always clear

Document-centric XML documents

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CLASS name="DCS317" num_of_std="100">
    <LECTURER lecid="111">Mounia</LECTURER>
    <STUDENT studid="007" >
        <NAME>James Bond</NAME> is the best student in the
        class. He scored <INTERM>95</INTERM> points out of
        <MAX>100</MAX>. His presentation of <ARTICLE>Using
        Materialized Views in Data Warehouse</ARTICLE> was
        brilliant.
    </STUDENT>
    <STUDENT stuid="131">
        <NAME>Donald Duck</NAME> is not a very good
        student. He scored <INTERM>20</INTERM> points…
        </STUDENT>
</CLASS>
```

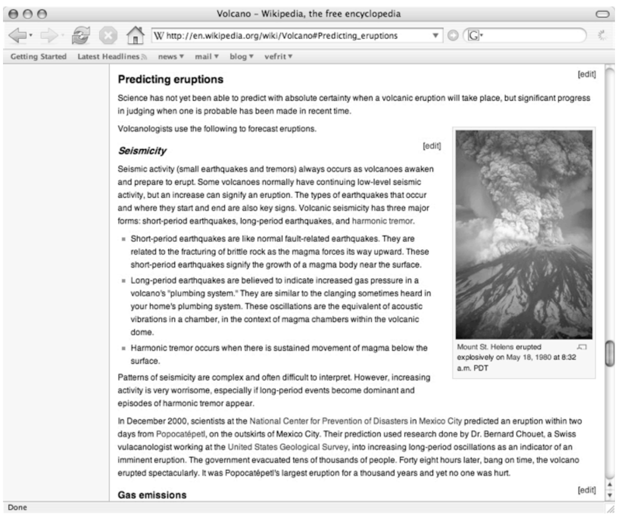
Focused retrieval: Scientific Collection

- Query
  - model checking aviation systems
- Answer
  - one section in a workshop report



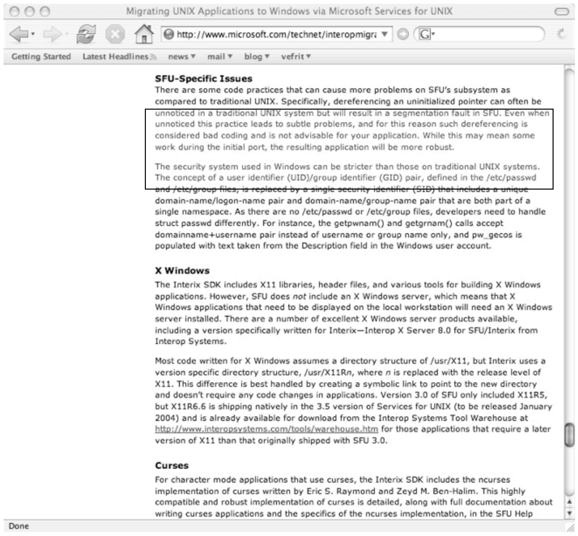
Focused Retrieval: Encyclopedia

- Information need
  - volcanic eruption prediction
- Answer
  - relatively small portion of the volcano topic



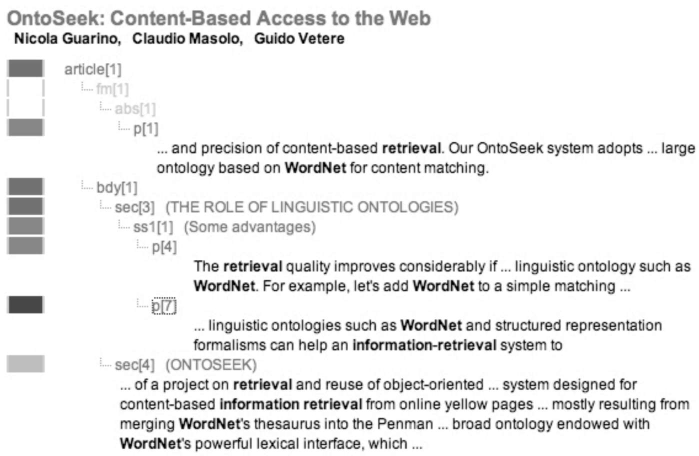
Focused retrieval: Technical Manual

- Query
  - segmentation fault windows services for unix
- Answer
  - only a single paragraph in a long manual



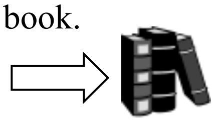
Focused retrieval: Right level of granularity

Query: wordnet information retrieval



Structured Document Retrieval (SDR)

- Traditional IR is about finding relevant documents to a user's information need, e.g. entire book.

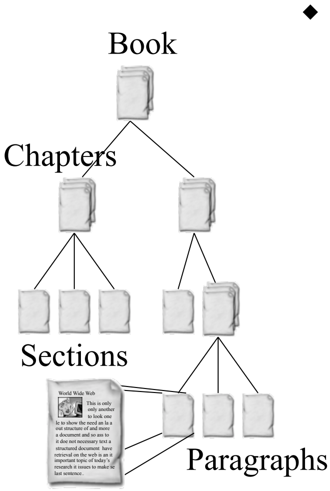


- SDR allows users to retrieve document components that are more focussed to their information needs, e.g a chapter of a book instead of an entire book.

- The structure of documents is exploited to identify which document components to retrieve.

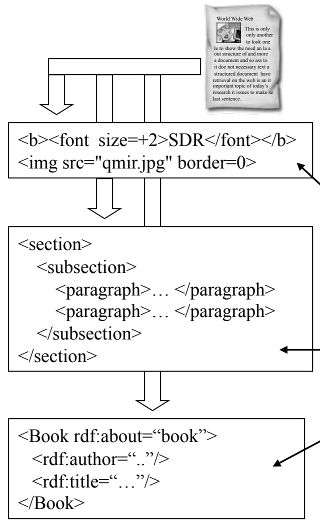
• Structure improves precision
• Exploit visual memory

Structured Documents



- In general, any document can be considered structured according to one or more structure-type
  - **Linear order of words, sentences, paragraphs …**
  - **Hierarchy or logical structure of a book's chapters, sections …**
  - Links (hyperlink), cross-references, citations …
  - Temporal and spatial relationships in multimedia documents

Structured Documents



- The structure can be implicit or explicit
- Explicit structure is formalised through document representation standards (Mark-up Languages)
  - Layout
    - LaTeX (publishing), HTML (Web publishing)
  - **Structure**
    - **SGML, XML (Web publishing, engineering), MPEG-7 (broadcasting)**
  - Content/Semantic
    - RDF (ontology)

Microformats

- Community data formats
  - Personal Data: hCard (vCard)
  - Calendar and Events: hCal (iCal)
  - Social Networking: XFN
  - Reviews: hReview
  - Licenses: rel-license
  - Folksonomies: rel-tag
- Embedded in XHTML pages and RSS feeds
  - Also RSS Extensions (iTunes, Yahoo! Media, Geo, Google Base, 20+ more in use)

Example: hCal

<strong class="summary">Fashion Expo in
Paris, France:
<abbr class="dtstart" title="2006-10-20">Oct 20</abbr>
to <abbr class="dtend" title="2006-10-23">22</abbr>

- Large and growing list of websites
  - Eventful.com
  - LinkedIn
  - Yedda
  - upcoming.yahoo.com
  - Yahoo! Local, Yahoo! Tech Reviews
- Benefit from shared tools, practices (hCalendar creator, iCal Extraction)

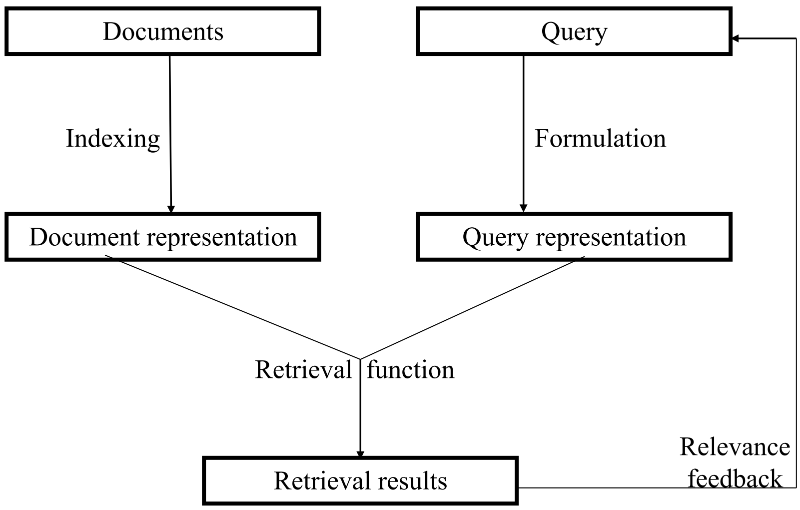
Queries in SDR

- Three types of queries:
  - Content-only (CO) queries
    - Standard IR queries but here we are retrieving document components
    - "London tube strikes"
  - Structure-only queries
    - Usually not that useful from an IR perspective
    - "Paragraph containing a diagram next to a table"

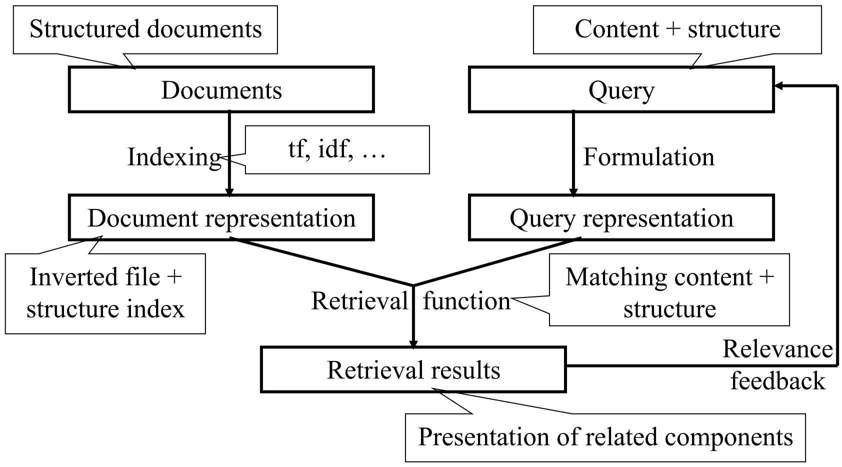
Queries in SDR

- Three types of queries:
  - Content-and-structure (CAS) queries
    - Put on constraints on which types of components are to be retrieved
      - E.g. "Sections of an article in the Times about congestion charges"
      - E.g. Articles that contain sections about congestion charges in London, and that contain a picture of Ken Livingstone, and return titles of these articles"
    - Inner constraints (support elements), target elements

Conceptual model for IR



Conceptual model for SDR



Conceptual model for SDR

Structured documents

XML is the currently adopted format for structured documents

tf, idf, …

e.g. agw can be used to capture the importance of the structure

Content + structure

query languages referring to content and structure are being developed for accessing XML documents, e.g. XIRQL, NEXI, XQueryFT

Inverted file + structure index

structure index captures in which document component the term occurs (e.g. title, section), as well as the type of document components (e.g. XML tags)

Matching content + structure

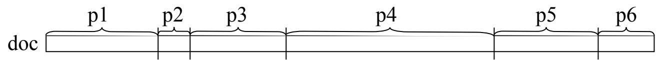
additional constraints are imposed from the structure

Presentation of related components

e.g. a chapter and its sections may be retrieved

Passage retrieval

- Passage: continuous part of a document,
  Document: set of passages



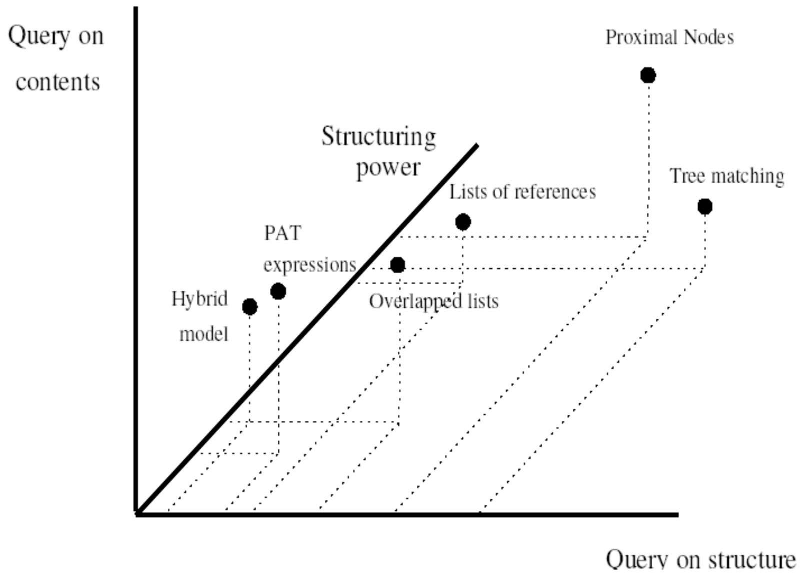
- A passage can be defined in several ways:
  - Fixed-length e.g. (300-word windows, overlapping)
  - Discourse (e.g. sentence, paragraph) ← e.g. according to logical structure but fixed (e.g. passage = sentence, or passage = paragraph)
  - Semantic (TextTiling based on sub-topics)
- Apply IR techniques to passages
  - Retrieve passage or document based on highest ranking passage or sum of ranking scores for all passages
  - Deal principally with content-only queries

(Callan, SIGIR 1994; Wilkinson, SIGIR 1994; Salton etal, SIGIR 1993; Hearst & Plaunt, SIGIR 1993; …)

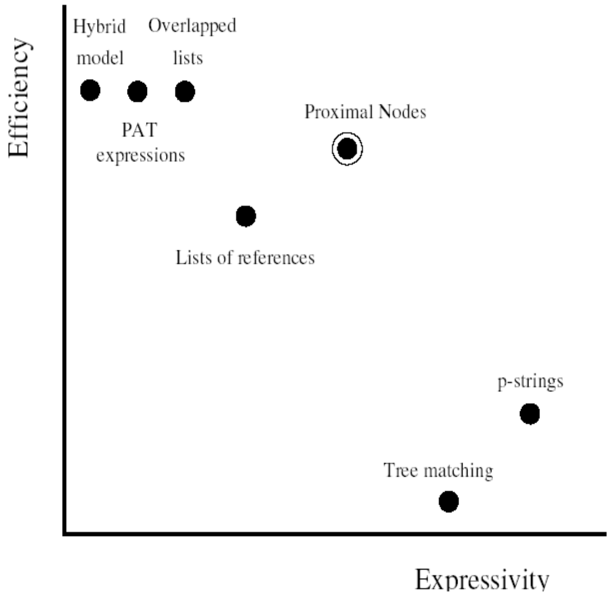
Structured document (text) retrieval

- Trade-off: expressiveness vs. efficiency
- Models (1989-1995)
  - Hybrid model (flat fields)
  - PAT expressions
  - Overlapped lists
  - Reference lists
  - Proximal nodes
  - Region algebra
    - Proposed as Algebra for XML-IR-DB Sandwich
  - p-strings
  - Tree matching

Comparison



Comparison



Example: Proximal Nodes

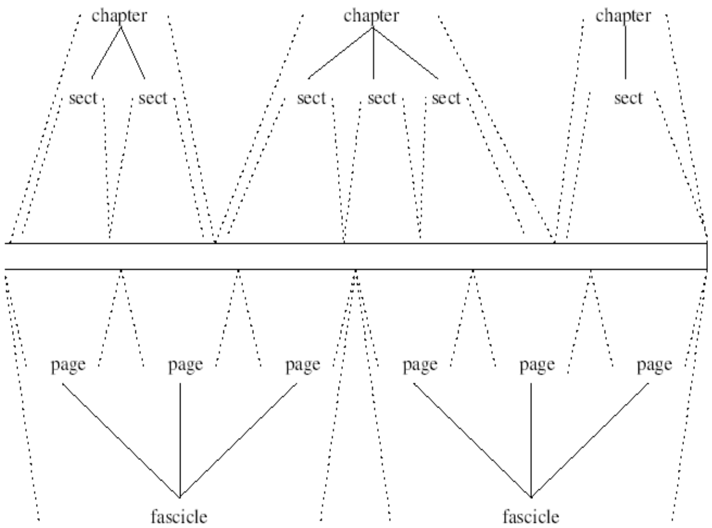
- Hierarchical structure
- Set-oriented language
- Avoid traversing the whole database
- Bottom-up strategy
- Solve leaves with indexes
- Operators work with near-by nodes
- Operators cannot use the text contents
- Most XPath and XQuery expressions can be solved using this model

(Navarro & Baeza-Yates, 1995)

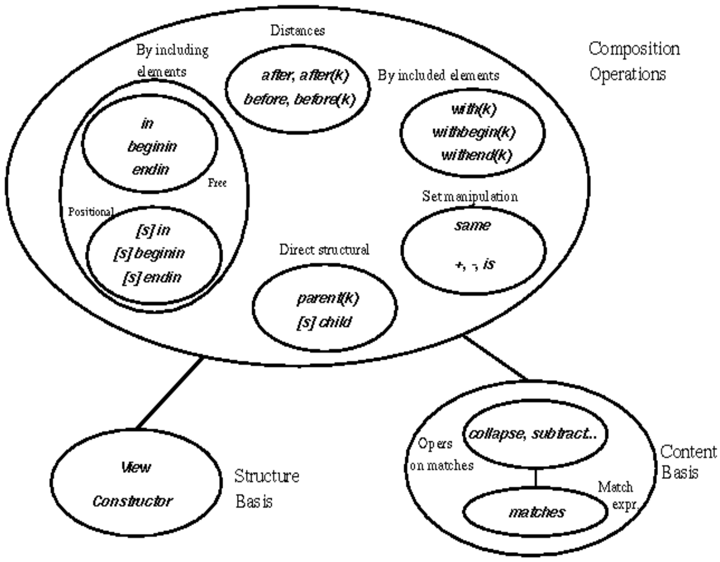
Proximal Nodes: Data Model

- Text = sequence of symbols (filtered)
- Structure = set of independent and disjoint hierarchies or “views”
- Node = Constructor + Segment
- Segment of node $\subseteq$ segment of children
- Text view, to modelize pattern-matching queries
- Query result = subset of some view

Proximal Nodes: Hierarchies

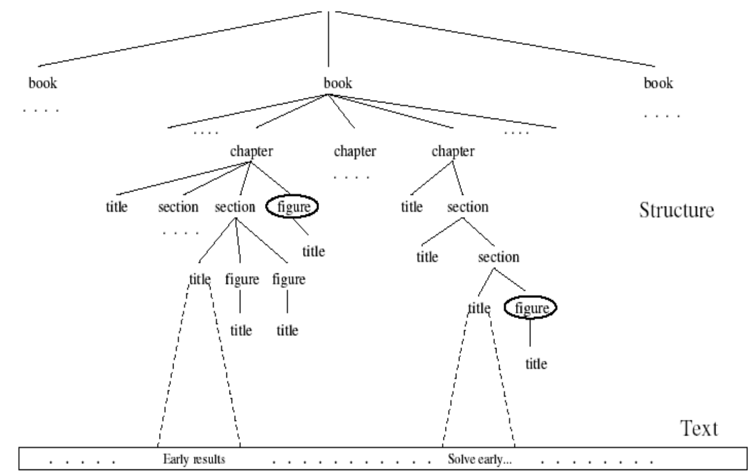


Proximal Nodes: Operations



Proximal Nodes: Query Example

`[last] figure in (chapter with (section with (title with "early")))`



Region algebra

- Manipulates text intervals - "between which positions in the document?"; and uses containment relationships - "in which components?"
  - Various methods but with similar aims: Simple Concordance List, Generalised Concordance List, Proximal Nodes …

100.1 100.2 101 102 103 103.1
<lecture> <title> Structured Document Retrieval </title>
103.2 103.3 104 105 106 107 108 108.1
<chapter> <title> Introduction into Structured Document Retrieval </title>
… SDR **…**
167.2 167.3
</chapter> </lecture>

  - Ranking based on word distances
  - Suited for CO and CAS queries

Query: "document" and "retrieval"
Intervals: {(102, 103)(107, 108)}

Query: [chapter] containing SDR
Intervals: {(103.2, 167.2)}

(SIGIR 1992, but see also XML retrieval Mihajlovic etal CIKM 2005)

Query languages for XML

- Four "levels" of expressiveness
  - Keyword search (CO Queries)
    - "xml"
  - Tag + Keyword search
    - book: xml
  - Path Expression + Keyword search (CAS Queries)
    - /book[./title about "xml db"]
  - XQuery + Complex full-text search
    - for $b in /book
let score $s := $b ftcontains "xml" && "db"
distance 5

Query languages for XML

- Keyword search (CO Queries)
  - "xml"
- Tag + Keyword search
  - book: xml
- Path Expression + Keyword search (CAS Queries)
  - /book[./title about "xml db"]
- XQuery + Complex full-text search
  - for $b in /book
let score $s := $b ftcontains "xml" && "db"
distance 5

XRank

```
<workshop date="28 July 2000">
    <title> XML and Information Retrieval: A SIGIR 2000 Workshop </title>
    <editors> David Carmel, Yoelle Maarek, Aya Soffer </editors>
    <proceedings>
        <paper id="1">
            <title> XQL and Proximal Nodes </title>
            <author> Ricardo Baeza-Yates </author>
            <author> Gonzalo Navarro </author>
            <abstract> We consider the recently proposed language … </abstract>
            <section name="Introduction">
                Searching on structured text is becoming more important with XML …
                <subsection name="Related Work">
                    The XQL language …
                </subsection>
            </section>
            …
            <cite xmlns:xlink="http://www.acm.org/www8/paper/xmlql> … </cite>
        </paper>
        …
```

(Guo etal, SIGMOD 2003)

XRank

```
<workshop date="28 July 2000">
    <title> XML and Information Retrieval: A SIGIR 2000 Workshop </title>
    <editors> David Carmel, Yoelle Maarek, Aya Soffer </editors>
    <proceedings>
        <paper id="1">
            <title> XQL and Proximal Nodes </title>
            <author> Ricardo Baeza-Yates </author>
            <author> Gonzalo Navarro </author>
            <abstract> We consider the recently proposed language … </abstract>
            <section name="Introduction">
                Searching on structured text is becoming more important with XML …
                <subsection name="Related Work">
                    The XQL language …
                </subsection>
            </section>
            …
            <cite xmlns:xlink="http://www.acm.org/www8/paper/xmlql> … </cite>
        </paper>
        …
```

XIRQL

```
<workshop date="28 July 2000">
   <title> XML and Information Retrieval: A SIGIR 2000 Workshop </title>
   <editors> David Carmel, Yoelle Maarek, Aya Soffer </editors>
   <proceedings>
      <paper id="1">
         <title> XQL and Proximal Nodes </title>
         <author> Ricardo Baeza-Yates </author>
         <author> Gonzalo Navarro </author>
         <abstract> We consider the recently proposed language … </abstract>
         <section name="Introduction">
            Searching on structured text is becoming more important with XML …
            <em> The XQL language </em>
         </section>
         …
         <cite xmlns:xlink="http://www.acm.org/www8/paper/xmlql> … </cite>
      </paper>
      …
```

index nodes (arrows pointing to <section> and <em>)

(Fuhr & Großjohann, SIGIR 2001)

Query languages for XML

- Keyword search (CO Queries)
  - "xml"
- Tag + Keyword search
  - book: xml
- Path Expression + Keyword search (CAS Queries)
  - /book[./title about "xml db"]
- XQuery + Complex full-text search
  - for $b in /book
    let score $s := $b ftcontains "xml" && "db"
    distance 5

XSearch

```
<workshop date="28 July 2000">
   <title> XML and Information Retrieval: A SIGIR 2000 Workshop </title>
   <editors> David Carmel, Yoelle Maarek, Aya Soffer </editors>
   <proceedings>
      <paper id="1">
         <title> XQL and Proximal Nodes </title>
         <author> Ricardo Baeza-Yates </author>
         <author> Gonzalo Navarro </author>
         <abstract> We consider the recently proposed language … </abstract>
         <section name="Introduction">
            Searching on structured text is becoming more important with XML …
            …
      </paper>
      <paper id="2">
         <title> XML Indexing </title>
         …
      <paper id="2">
```

Not a "meaningful" result

(Cohen etal, VLDB 2003)

Query languages for XML

- Keyword search (CO Queries)
  - "xml"
- Tag + Keyword search
  - book: xml
- Path Expression + Keyword search (CAS Queries)
  - /book[./title about "xml db"]
- XQuery + Complex full-text search
  - for $b in /book
    let score $s := $b ftcontains "xml" && "db"
    distance 5

XPath

- fn:contains($e, string) returns true iff $e contains string

  //section[fn:contains(./title, "XML Indexing")]

(W3C 2005)

XIRQL

- Weighted extension to XQL (precursor to XPath)

  //section[0.6 · .//* cw "XQL" +
  0.4 · .//section cw "syntax"]

(Fuhr & Großjohann, SIGIR 2001)

XXL

- Introduces similarity operator ~

```
Select  Z
From   http://www.myzoos.edu/zoos.html
Where  zoos.#.zoo As Z and
        Z.animals.(animal)?.specimen as A and
        A.species ~ “lion” and
        A.birthplace.#.country as B and
        A.region ~ B.content
```

(Theobald & Weikum, EDBT 2002)

NEXI

- Narrowed Extended XPath I
- INEX Content-and-Structure (CAS) Queries
- Specifically targeted for content-oriented XML search (i.e. “aboutness”)

```
//article[about(.//title, apple) and
          about(.//sec, computer)]
```

(Trotman & Sigurbjornsson, INEX 2004)

Query languages for XML

- Keyword search (CO Queries)
  - "xml"
- Tag + Keyword search
  - book: xml
- Path Expression + Keyword search (CAS Queries)
  - /book[./title about "xml db"]
- XQuery + Complex full-text search
  - for $b in /book
    let score $s := $b ftcontains "xml" && "db"
    distance 5

Schema-Free XQuery

- Meaningful least common ancestor (mlcas)

```
for $a in doc("bib.xml")//author
    $b in doc("bib.xml")//title
    $c in doc("bib.xml")//year
where $a/text() = "Mary" and
        exists mlcas($a,$b,$c)
return <result> {$b,$c} </result>
```

(Li etal, VLDB 2003)

XQuery Full-Text

- Two new XQuery constructs

1) FTContainsExpr
   - Expresses "Boolean" full-text search predicates
   - Seamlessly composes with other XQuery expressions

3) FTScoreClause
   - Extension to FLWOR expression
   - Can score FTContainsExpr *and* other expressions

(W3C 2005)

FTContainsExpr

//book ftcontains "Usability" && "testing" distance 5

//book[./content ftcontains "Usability" with stems]/title

//book ftcontains /article[author="Dawkins"]/title

FTScore Clause

In any order {

```
FOR  $v [SCORE $s]? IN [FUZZY] Expr
LET   …
WHERE …
ORDER BY …
RETURN
```

Example

```
FOR $b SCORE $s in
                /pub/book[. ftcontains "Usability" && "testing"
                        and ./price < 10.00]
ORDER BY $s
RETURN $b
```

FTScore Clause

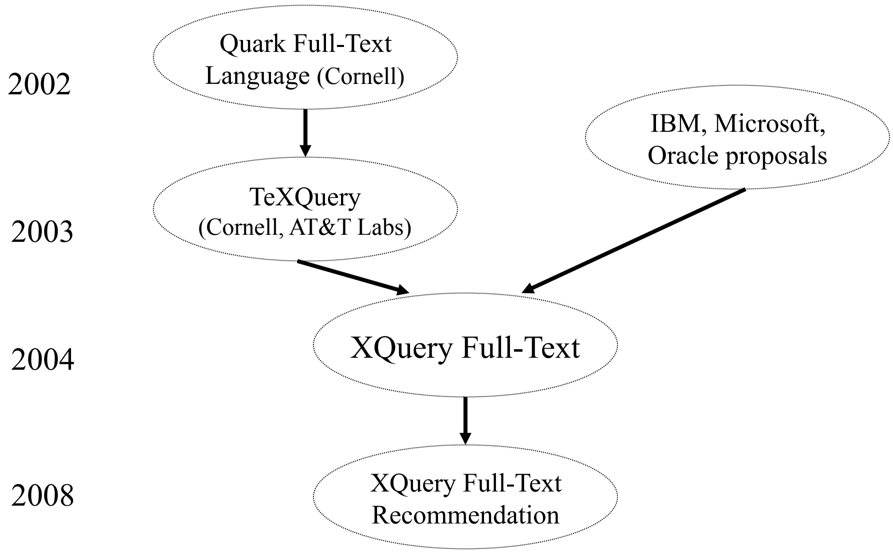
In any order {

```
FOR  $v [SCORE $s]? IN [FUZZY] Expr
LET   …
WHERE …
ORDER BY …
RETURN
```

Example

```
FOR $b SCORE $s in FUZZY
                /pub/book[. ftcontains "Usability" && "testing"]
ORDER BY $s
RETURN $b
```

XQuery Full-Text Evolution



Query languages for XML - Recap

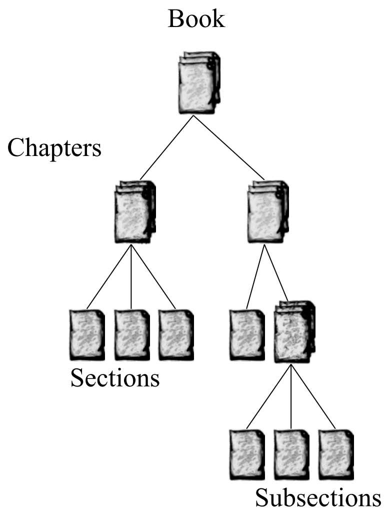
- Virtues and setbacks of XML query languages
  - Expressive query languages
  - But, too complex for many applications
  - Different interpretations

Element retrieval

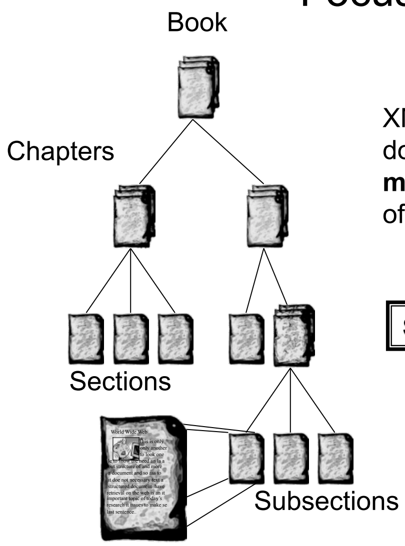
- XML retrieval vs. document retrieval
- XML retrieval = **Focused** retrieval
- Challenges
  1. Term statistics
  2. Relationship statistics
  3. Structure statistics
  4. Overlapping elements
  5. Interpretations of structural constraints
- Ranking
  1. Retrieval units
  2. Combination of evidence
  3. Post-processing – Presentation of XML search results

XML retrieval vs. document retrieval

- No predefined unit of retrieval
- Dependency of retrieval units
- Aims of XML retrieval:
  - Not only to find relevant elements
  - But those at the appropriate level of granularity



Content-oriented XML retrieval = Focused Retrieval



XML retrieval allows users to retrieve document components that are **more focused**, e.g. a subsection of a book instead of an entire book.

SEARCHING = QUERYING + BROWSING

Note:
Here, document component = XML element

Focused Retrieval for XML: Principle

- A XML retrieval system should always retrieve the most specific part of a document answering a query.

- Example query: **football**
- Document

```
<chapter> 0.3 football
  <section> 0.5 history </section>
  <section> 0.8 football 0.7 regulation </section>
</chapter>
```

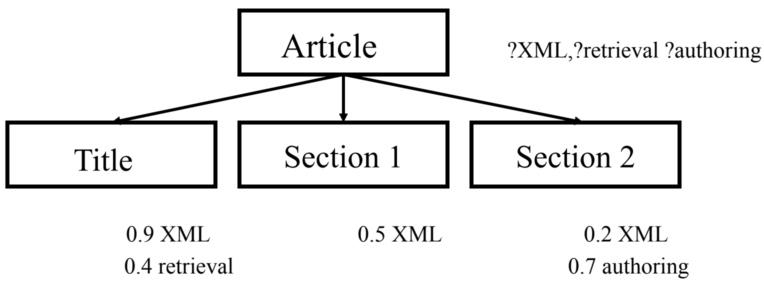
- Return <section>, not <chapter>

Content-oriented XML retrieval = Focused Retrieval

Return document components of **varying granularity** (e.g. a book, a chapter, a section, a paragraph, a table, a figure, etc), relevant to the user's information need both with regards to **content** and **structure**.

SEARCHING = QUERYING + BROWSING

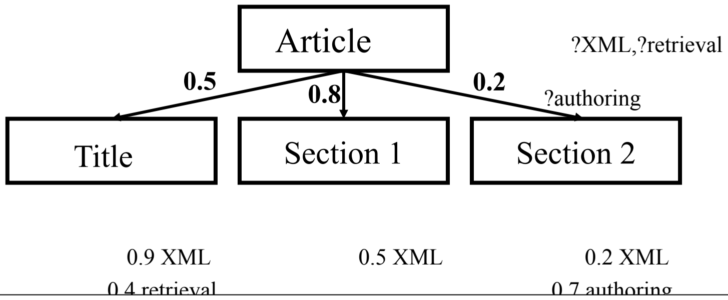
Challenge 1: Term statistics



No fixed retrieval unit + nested document components:

- ❑ how to obtain element and collection statistics (e.g. tf, idf)?
- ❑ which aggregation formalism to use?
- ❑ inner or outer aggregation?

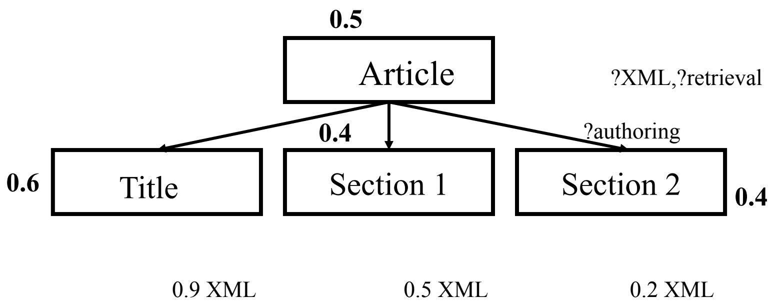
Challenge 2: Relationship statistics



Relationship between elements:

- which sub-element(s) contribute best to content of its parent element and vice versa?
- how to estimate (or learn) relationship statistics (e.g. size, number of children, depth, distance)?
- how to aggregate term and/or relationship statistics?

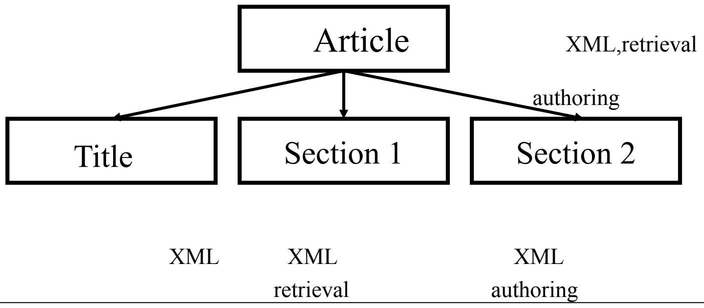
Challenge 3: Structure statistics



Different types of elements:

- which element is a good retrieval unit?
- is element size an issue?
- how to estimate (or learn) structure statistics (frequency, user studies, size, depth)?
- how to aggregate term, relationship and/or structure statistics?

Challenge 4: Overlapping elements



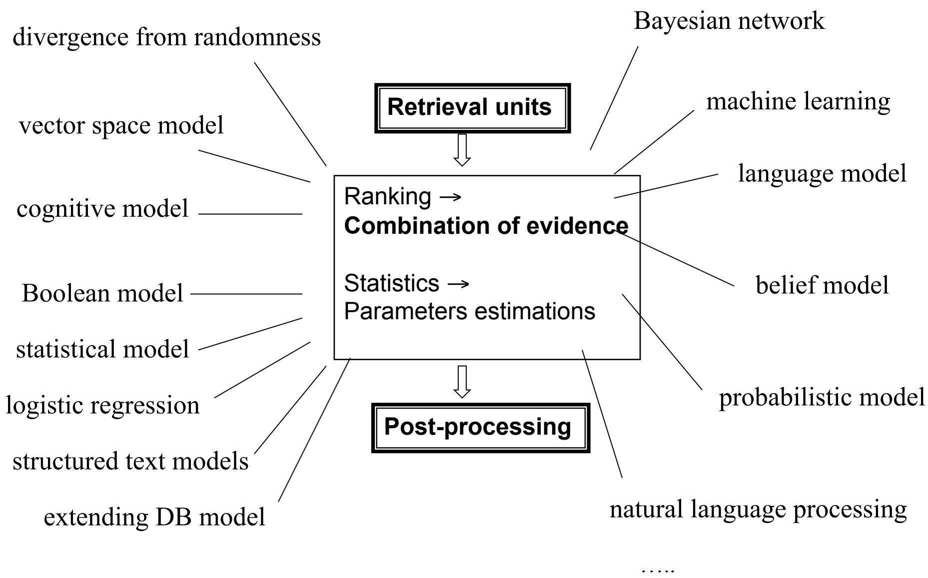
Nested (overlapping) elements:

- section 1 and article are both relevant to "XML retrieval"
- which one to return so that to reduce overlap?
- should the decision be based on user studies, size, types, etc?

Challenge 5: Expressing and interpreting structural constraints

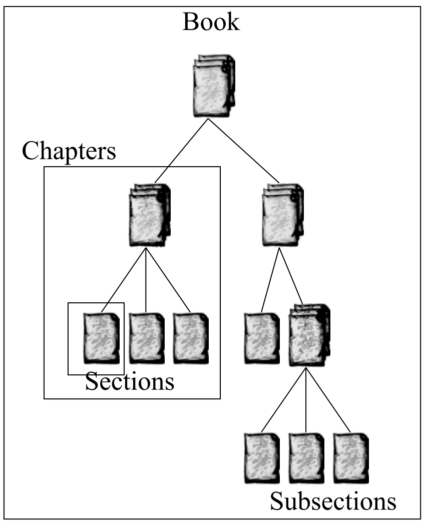
- Ideally:
  - There is one DTD/schema
  - User understands DTD/schema
- In practice: rare
  - Many DTs/schemas
  - DTDs/Schema not known in advance
  - DTDs/Schema change
  - Users do not understand DTDs/schema
- Need to identify "similar/synonym" elements/tags
- Importance (weight) of tags
- Strict or vague interpretation of the structure
- Relevance feedback/blind feedback?

Retrieval models …



Retrieval units: What to Index?

- XML documents are trees
  - hierarchical structure of nested elements (sub-trees)
- What should we put in the index?
  - there is no fixed unit of retrieval



Retrieval units: XML sub-trees

Assume a document like

```
<article>
 <title>XXX</title>
   <abstract>YYY</abstract>
 <body>
   <sec>ZZZ</sec>
   <sec>ZZZ</sec>
 </body>
</article>
```

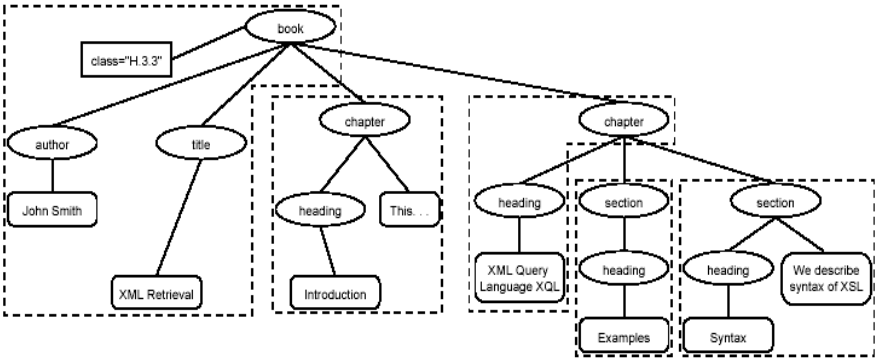
Index separately

```
• <article>XXX YYY ZZZ ZZZ </article>
• <title>XXX</title>
• <abstract>YYY</abstract>
• <body>ZZZ ZZZ</body>
• <sec>ZZZ</sec>
• <sec>ZZZ</sec>
```

Retrieval units: XML sub-trees

- Indexing sub-trees is closest to traditional IR
  - each XML elements is bag of words of itself and its descendants
  - and can be scored as ordinary plain text document
- Advantage: well-understood problem
- Negative:
  - redundancy in index
  - terms statistics
  - Led to the notion of indexing nodes
  - Problem: how to select them?
    - manually, frequency, relevance data

(XIRQL) Indexing nodes



(Fuhr & Großjohann, SIGIR 2001)

Retrieval units: Disjoint elements

Assume a document like

```
<article>
 <title>XXX</title>
   <abstract>YYY</abstract>
 <body>
   <sec>ZZZ</sec>
   <sec>ZZZ</sec>
 </body>
</article>
```

Index separately

- <title>XXX</title>
- <abstract>YYY</abstract>
- <sec>ZZZ</sec>
- <sec>ZZZ</sec>

Note that <body> and <article> have not been indexed

Retrieval units 2: Disjoint elements

- Main advantage and main problem
  - (most) article text is not indexed under /article
  - avoids redundancy in the index
- But how to score higher level (non-leaf) elements?
  - Propagation/Augmentation approach
  - Element specific language models

(Geva, INEX 2004, INEX 2005)

Propagation - GPX model

Leaf elements score

$$L = N^{n-1} \sum_{i=1}^{n} \frac{t_i}{f_i}$$

- n : the number of unique query terms
- N: a small integer (N=5, but any 10 > N >2 works)
- t_i : the frequency of the term in the leaf element
- f_i : the frequency of the term in the collection

Branch elements score

$$RSV = D(n) \sum_{i=1}^{n} L_i$$

- n : the number of children elements
- $D(n)$ = 0.49 if n = 1
  - 0.99 Otherwise
- **D(n) = relationship statistics**
- L_i : child element score
  - scores are recursively propagated up the tree

Element specific language model (simplified)

Assume a document
<bdy>
<sec>cat…</sec>
<sec>dog…</sec>
</bdy>

Query: cat dog
(Ogilvie & Callan, INEX 2004)

- Assume
  - P(dog|bdy/sec[1])=0.7
  - P(cat|bdy/sec[1])=0.3
  - P(dog|bdy/sec[2])=0.3
  - P(cat|bdy/sec[2])=0.7
- Mixture $P(w|e) = \sum \lambda_i P(w|e_i)$
  - With uniform weights (λ=0.5)
  - **λ = relationship statistics**
  - P(cat|bdy)=0.5
  - P(dog|bdy)=0.5
  - So /bdy will be returned

Retrieval units: Distributed

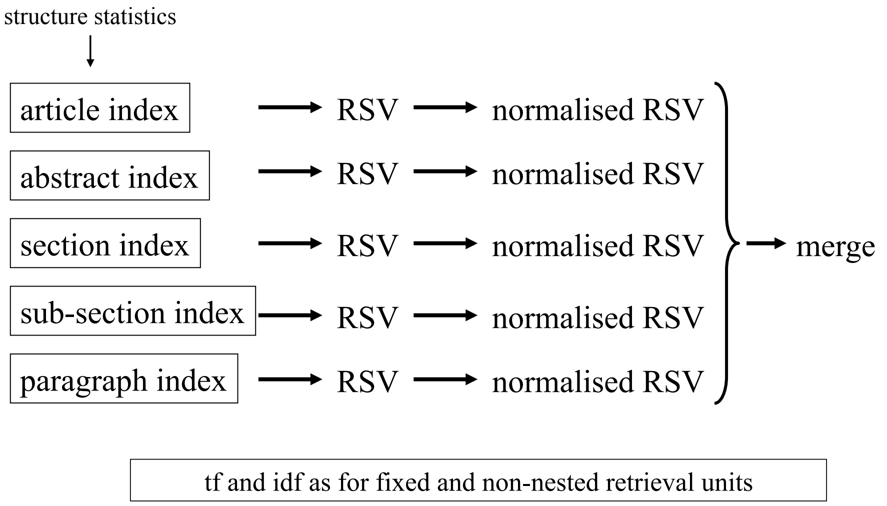
- Index separately particular types of elements
- E.g., create separate indexes for

articles
abstracts
sections
subsections
subsubsections
paragraphs …

} structure statistics

- Each index provides statistics tailored to particular types of elements
  - language statistics may deviate significantly
  - queries issued to all indexes
  - results of each index are combined (after score normalization)

Distributed: Vector space model



(Mass & Mandelbrod, INEX 2004)

Retrieval units: Distributed

- Only part of the structure is used
  - Element size
  - Relevance assessment
  - Others
- Main advantages compared to disjoint element strategy:
  - avoids score propagation which is expensive at run-time
  - index redundancy is basically pre-computing propagation
  - XML specific propagation requires nontrivial parameters to train
- Indexing methods and retrieval models are "standard" IR
  - although issue of merging - normalization

Combination: Language model

element language model
collection language model
smoothing parameter λ
⇒ element score

high value of λ leads to increase in size of retrieved elements

element size
element score
article score
⇒ rank element

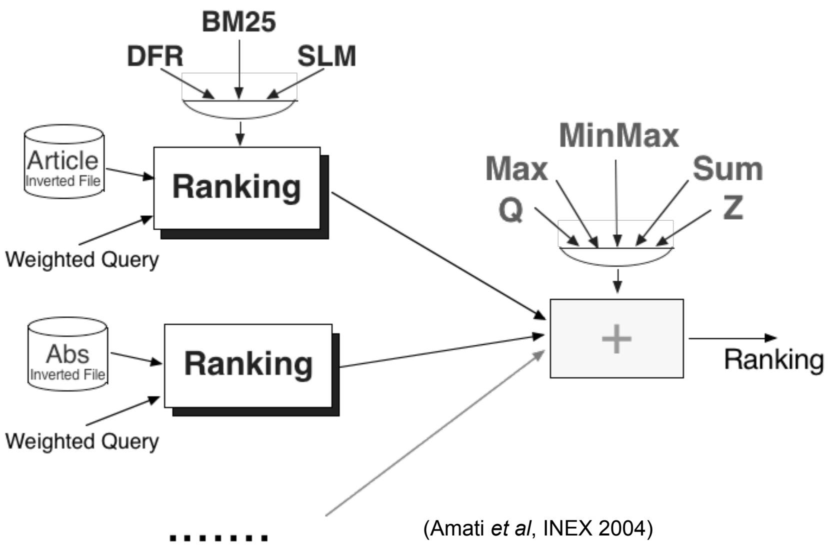
structure statistics

query expansion with blind feedback
ignore elements with ≤ 20 terms

relationship statistics

(Sigurbjörnsson etal, INEX 2003, INEX 2004)

Combination: Normalization



(Amati *et al*, INEX 2004)

Combination: Machine learning

- Use of standard machine learning to train a function that combines

  Parameter for a given element type
  Parameter * score(element)
  Parameter * score(parent(element))
  Parameter * score (document)

  structure statistics
  relationship statistics

- Training done on relevance data (previous years)
- Scoring done using OKAPI

(Vittaut & Gallinari, ECIR 2006)

Combination: Contextualization

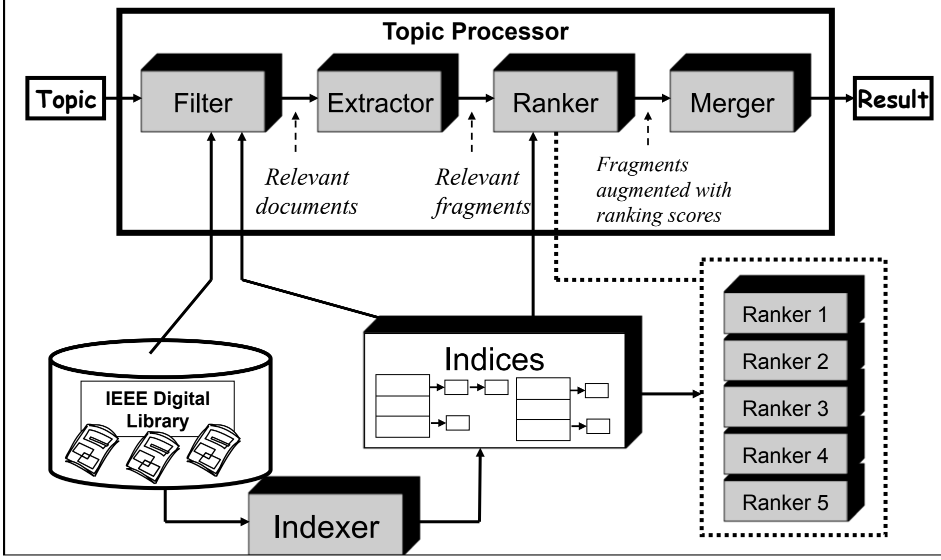
- Basic ranking by adding weight value of all query terms in element.
- Re-weighting is based on the idea of using the ancestors of an element as a context.
  - Root: combination of the weight of an element its 1.5 * root.
  - Parent: average of the weights of the element and its parent.
  - Tower: average of the weights of an element and all its ancestors.
  - Root + Tower: as above but with 2 * root.
- Here root is the document

(Arvola et*al*, CIKM 2005, INEX 2005)

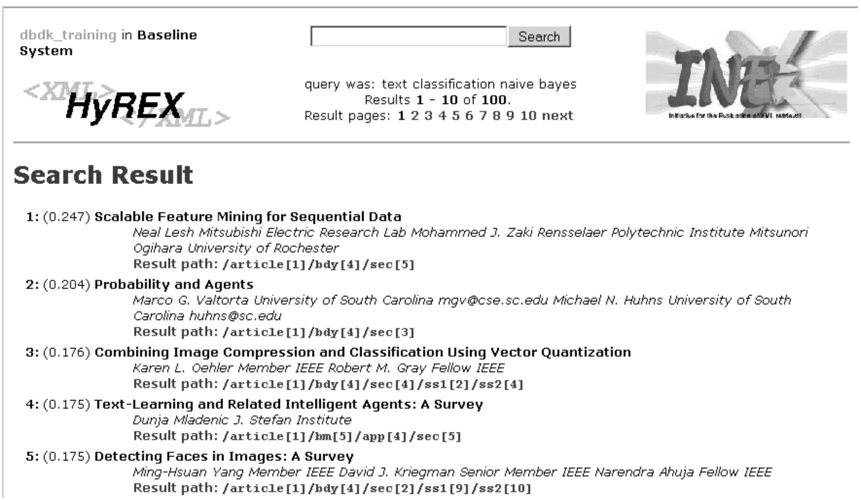
Combination - Merging

(Ben-Aharon, INEX 2003)

–Word Number
–IDF
–Similarity
–Proximity
–TFIDF



Presenting XML retrieval results



(Goevert etal, INEX 2002)

Presenting XML retrieval results

- XML element retrieval is a core task
  - how to estimate the relevance of individual elements
- However, it may not be the end task
  - Simply returning a ranked list of elements results seems insufficient
    - **remove or reduce overlapping elements**
    - elements from the same article may be grouped
    - return one element per article (best entry point from where start to read relevant content)
- Note
  - Presentation and interface

New retrieval tasks (at INEX)

- INEX 2005 … addressed new retrieval tasks
  - Thorough is 'pure' XML element retrieval as before
  - **Focused does not allow for overlapping elements to be returned**
  - **Fetch and Browse requires results to be clustered per article**
    - Various variants
  - **Passage retrieval**
- New tasks require post-processing of 'pure' XML element runs
  - geared toward displaying them in a particular interface

Post-processing: Controlling Overlap

What most approaches are doing:

- Given a ranked list of elements:

1. select element with the highest score within a path
2. discard all ancestors and descendants
3. go to step 1 until all elements have been dealt with

- (Also referred to as brute-force filtering)

"Post"-Processing: Removing overlap

- Sometimes with some "prior" processing to affect ranking:
  - Use of a utility function that captures the amount of useful information in an element

Element score * Element size * Amount of relevant information

  - Used as a prior probability
  - Then apply "brute-force" overlap removal

(Mihajlovic etal, INEX 2005; Ramirez etal, FQAS 2006))

Post-processing: Controlling Overlap

- Start with a component ranking, elements are re-ranked to control overlap.
- Retrieval status values of those components containing or contained within higher ranking components are iteratively adjusted
- (depends on amount of overlap "allowed")

1. Select the highest ranking component.
2. Adjust the retrieval status value of the other components.
3. Repeat steps 1 and 2 until the top *m* components have been selected.

(Clarke, SIGIR 2005)

Post-Processing: Removing overlap

(Mass & Mandelbrod, INEX 2005)

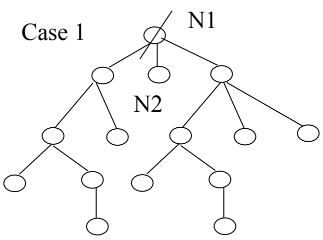
Smart filtering

Given a list of rank elements

-group elements per article

-build a result tree

-"score grouping":

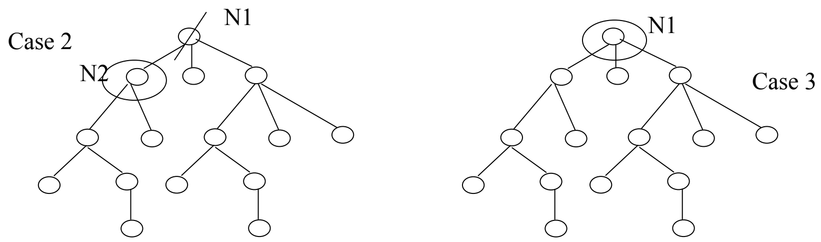


-for each element N1

1. score N2 > score N1

2. concentration of good elements

3. even distribution of good elements



CAS query processing: sub-queries

- Sub-queries decomposition
  - *//article [search engines] // sec [Internet growth] AND sec [Yahoo]*
    - article [search engines]
    - sec [Internet growth]
    - sec [Yahoo]
- Run each sub-queries and then combine
- Reward structure matching (strict vs vague)

(Sauvagnat etal, INEX 2005)

Example of combination: Probabilistic algebra

// article [about(.,bayesian networks)] // sec [about(., learning structure)]

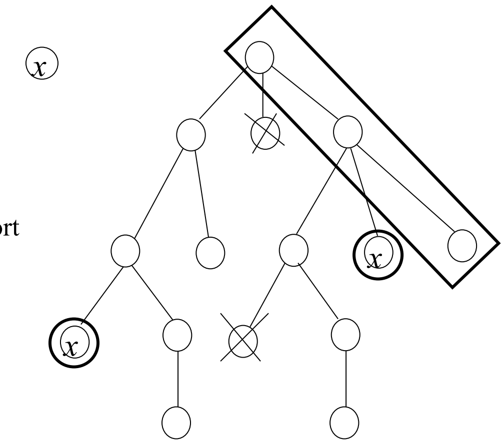
$$R(\text{learning structure}) \cap \text{label}^{-1}(\text{sec}) \cap \text{descendants}\left(R(\text{bayesian networks}) \cap \text{label}^{-1}(\text{article})\right)$$

- “Vague” sets
  - R(…) defines a vague set of elements
  - label^{-1}(…) can be defined for strict or vague interpretation
- Intersections and Unions are computed as probabilistic “and” and fuzzy-or.

(Vittaut etal, INEX 2004)

CAS query processing: target and support matching

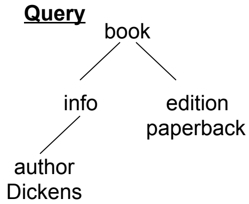
1. Generate tree according to target element's content and structural constraint.
2. Discard all target elements that do not fulfill structural ancestor and descendant constraints.
3. Do same as above but for all support element .
4. Collect elements left in step 3, and discard all of the target elements, which do not have such ancestor elements.



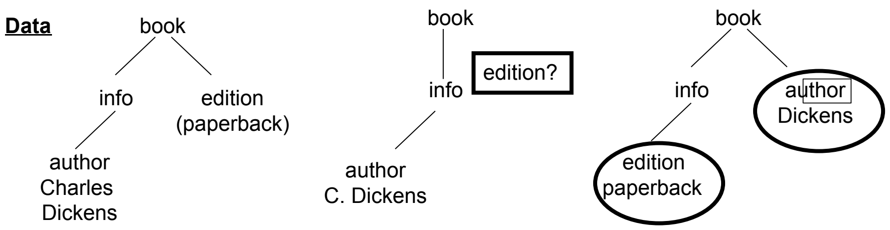
Vague structural constraints

- Define score between two tags/paths
- Boost content score with tag/path score
- Use of dictionary of equivalent tags/synonym list
  - Analysis of the collection DTD
    - Syntactic, e.g. "p" and "ip1"
    - Semantic, e.g. "capital" and "city"
  - Analysis of past relevance assessments
    - For topic on "section" element, all types of elements assessed relevant added to "section" synonym list
    - Probabilistic estimation of tag weights
- Ignore structural constraint for target, support element or both
- Relaxation techniques from DB (e.g. lowest common ancestor, etc)

XML Query Relaxation (FleXPath)
where DB and IR meet

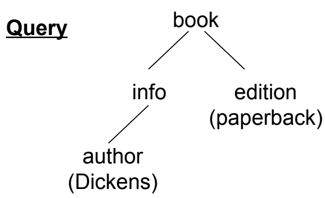


- Tree pattern relaxations:
  - Leaf node deletion
  - Edge generalization
  - Subtree promotion

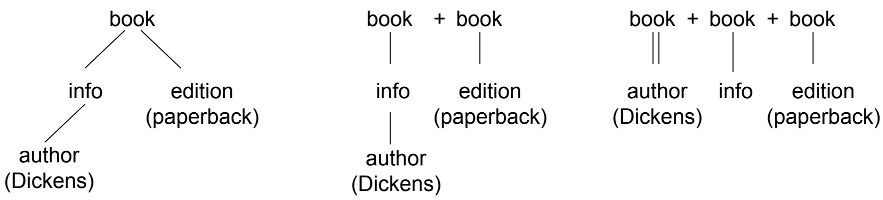


(Amer-Yahia, SIGMOD 2004) (Schlieder, EDBT 2002)
(Delobel & Rousset, 2002) (Amer-Yahia etal, VLDB 2005)

A Family of XML Scoring Methods



- Twig scoring
  - High quality
  - Expensive computation
- Path scoring
- Binary scoring
  - Low quality
  - Fast computation



(Amer-Yahia, VLDB 2005)

XML Element retrieval - Recap

- Choice of retrieval units can affect the "type" of retrieval models
- XML retrieval can be viewed as a **combination of evidence** problem
- No "clear winner" in terms of retrieval models
  - We still miss the benchmark/baseline approach
  - Lots of heuristics
- BUT WHAT SEEM TO WORK WELL:
  - Element
  - Document
  - Size
- ***Thorough investigation for all ranking models, all indexing approaches, and all evidence needed***

User aspects

- User study - INEX interactive track
- Incorporating user behaviour

Evaluation of XML retrieval: INEX

- Evaluating the effectiveness of **content-oriented** XML retrieval approaches

- Similar methodology as for TREC, but adapted to XML retrieval

  (to be described later)

DELOS
Network of Excellence on Digital Libraries

Interactive Track in 2004

- **Investigate behaviour of searchers when interacting with XML components**
- Content-only Topics
  - topic type an additional source of context
    - Background topics / Comparison topics
  - 2 topic types, 2 topics per type
  - 2004 INEX topics have added task information
- Searchers
  - "distributed" design, with searchers spread across participating sites

Topic Example

<title>+new +Fortran +90 +compiler**</title>**

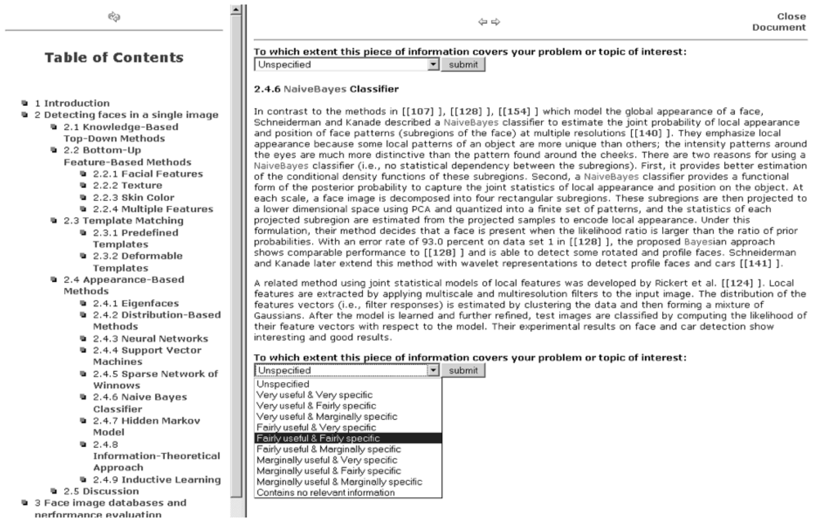
<description> How does a Fortran 90 compiler differ from a compiler for the Fortran before it. **</description>**

<narrative> I've been asked to make my Fortran compiler compatible with Fortran 90 so I'm interested in the features Fortran 90 added to the Fortran standard before it. I'd like to know about compilers (they would have been new when they were introduced), especially compilers whose source code might be available. Discussion of people's experience with these features when they were new to them is also relevant. An element will be judged as relevant if it discusses features that Fortran 90 added to Fortran. **</narrative>**

<keywords>new Fortran 90 compiler**</keywords>**

Baseline system

Baseline system



Some results

- How far down the ranked list?
  - 83 % from rank 1-10
  - 10 % from rank 11-20
- Query operators rarely used
- 80 % of queries consisted of 2, 3, or 4 words
- Accessing components
  - ~2/3 was from the ranked list
  - ~1/3 was from the document structure (ToC)
- 1st viewed component from the ranked list
  - 40% article level, 36% section level, 22% ss1 level, 4% ss2 level
- ~ 70 % only accessed 1 component per document

Document-centric XML retrieval: Conclusions

- SDR → now mostly about XML retrieval
- Efficiency:
  - Not just documents, but all its elements
- Models
  - Units
  - Statistics
  - Combination
- User tasks
- Link to web retrieval / novelty retrieval
- Interface and visualisation
- Clustering, categorisation, summarisation

Outline

- Introduction to XML, basics and standards
- Document-oriented XML retrieval
- **Evaluating XML retrieval effectiveness**
- Going beyond XML retrieval

Evaluating XML retrieval effectiveness

- Structured document retrieval and evaluation
- XML retrieval evaluation
  - Collections
  - Topics
  - Retrieval tasks
  - Relevance and assessment procedures
  - Metrics
- INEX tracks

SDR and Evaluation

- Passage retrieval
  - Test collection built for that purpose, where passages in relevant documents were assessed (Wilkinson SIGIR 1994)
- Structured document retrieval
  - Web retrieval collection (museum) (Lalmas & Moutogianni, RIAO 2000)
  - Fictitious collection (Roelleke etal, ECIR 2002; Ruthven & Lalmas JDoc 1998)
  - Shakespeare collection (Kazai *et al*, ECIR 2003)
- **INEX initiative** (Kazai *et al*, JASIST 2004; INEX proceedings; SIGIR forum reports, …)
  - "Real" large test collection following TREC methodology
  - Evaluation campaign
  - XML

Evaluation of XML retrieval: INEX

- Evaluating the effectiveness of **content-oriented** XML retrieval approaches
- **Collaborative** effort ⇒ participants contribute to the development of the collection
  - queries
  - relevance assessments
  - methodology
- Similar methodology as for TREC, but adapted to XML retrieval

http://inex.is.informatik.uni-duisburg.de/

Document collections

Year	number documents	number elements	size	average number elements	average element depth	
2002-2004	12,107	8M	494MB	1,532	6.9	**IEEE**
2005	16,819	11M	764MB	‘’	‘’	
2006-2007	659,388	52M	60 (4.6) GB	161.35	6.72	**Wikipedia**

(Denoyer & Gallinari, SIGIR Forum, June 2006)

INEX 2009: a new larger collection, based on Wikipedia, with richer set of tags

Topics

In IR (TREC - http://trec.nist.gov/) evaluation, topics are made of:

- **Title field:**
  - short explanation of the information need.
- **Description field:**
  - one or two sentence natural language definition of the information need.
- **Narrative field:**
  - detailed explanation of information need
  - description of what makes something relevant
  - work task it might help to solve

- Keywords obtained during collection exploration for the topic creation
- On and off- topic keywords (Amitay *et al*, SIGIR 2004)

Two types of topics

- Content-only (**CO**) topics
  - ignore document structure
  - simulates users, who do not have any knowledge of the document structure or who choose not to use such knowledge
- Content-and-structure (**CAS**) topics
  - contain conditions referring both to content and structure of the sought elements
  - simulate users who do have some knowledge of the structure of the searched collection

CO topics

<title>
"Information Exchange", +"XML", "Information Integration"
</title>
<description>
How to use XML to solve the information exchange (information integration) problem, especially in heterogeneous data sources?
</description>
<narrative>
Relevant documents/components must talk about techniques of using XML to solve information exchange (information integration) among heterogeneous data sources where the structures of participating data sources are different although they might use the same ontologies about the same content.
</narrative>

CAS topics

<title>
//article[(./fm//yr = '2000' OR ./fm//yr = '1999') AND about(., '"intelligent transportation system"')]//sec[about(.,'automation +vehicle')]
</title>
<description>
Automated vehicle applications in articles from 1999 or 2000 about intelligent transportation systems.
</description>
<narrative>
To be relevant, the target component must be from an article on intelligent transportation systems published in 1999 or 2000 and must include a section which discusses automated vehicle applications, proposed or implemented, in an intelligent transportation system.
</narrative>

NEXI

- Narrowed Extended XPath I
- INEX Content-and-Structure (CAS) Queries
- Specifically targeted for content-oriented XML search (i.e. "**aboutness**")

//article[about(.//title, apple) and about(.//sec, computer)]

(Trotman & Sigurbjörnsson, INEX 2004)
(Sigurbjörnsson & Trotman, INEX 2003)

How to interpret structural constraints?

- Strict vs. vague interpretation of the structure led to:
  - CO+S topics
  - CAS topics

defined in INEX 2005

support element

CAS parent topic

target element

<title></title>
<castitle>//**article**[about(.,interconnected networks)]//**p**[about(., Crossbar networks)]</castitle>
<description>We are looking for paragraphs that talk about Crossbar networks from articles that talk about interconnected networks.</description>
<narrative>With networking between processors gaining significance, interconnected networks has become an important concept. Crossbar network is one of the interconnected networks. We are looking for information on what crossbar networks exactly are, how they operate and why they are used to connect processors. Any article discussing interconnected networks in the context of crossbar networks is considered to be relevant. Articles talking about interconnected networks such as Omega networks are not considered to be relevant. This information would be used to prepare a presentation for a lecture on the topic, and hence information on crossbar networks makes an element relevant.
</narrative>

CAS child topic

<castitle>//article//p[about(., Crossbar networks)]</castitle>
<parent>//article[about(.,interconnected networks)]//p[about(., Crossbar networks)]
</parent>
<description>We are looking for paragraphs that talk about Crossba networks.
</description>
<narrative>With networking between processors gaining significance, interconnected networks has become an important concept. Crossbar network is one of the interconnected networks. We are looking for information on what crossbar networks exactly are, how they operate and why they are used to connect processors. Any paragraph discussing interconnected networks in the context of crossbar networks is considered to be relevant. Articles talking about interconnected networks such as Omega networks are not considered to be relevant. This information would be used to prepare a presentation for a lecture on the topic, and hence information on crossbar networks makes an element relevant.
</narrative>

CO+S topics

<title>markov chains in graph related algorithms</title>
<castitle>//article//sec[about(.,+"markov chains" +algorithm +graphs)] </castitle>
<description>Retrieve information about the use of markov chains in graph theory and in graphs-related algorithms.
</description>
<narrative>I have just finished my Msc. in mathematics, in the field of stochastic processes. My research was in a subject related to Markov chains. My aim is to find possible implementations of my knowledge in current research. I'm mainly interested in applications in graph theory, that is, algorithms related to graphs that use the theory of markov chains. I'm interested in at least a short specification of the nature of implementation (e.g. what is the exact theory used, and to which purpose), hence the relevant elements should be sections, paragraphs or even abstracts of documents, but in any case, should be part of the content of the document (as opposed to, say, vt, or bib).
</narrative>

Impact of structural constraints in queries

- Make use of CO+**S topics**: <castitle>
- **Structural hints**:
  - “Upon discovering that his/her <title> query returned many irrelevant elements, a user might decide to add structural hints, i.e. to write his/her initial CO query as a CAS query”

open standards for digital video in distance learning

⇩

//article//sec[about(.,**open standards for digital video in distance learning**)]

- Results show that processing the structure (from the query) does not consistently lead to any significant improvement in retrieval effectiveness (apart for maybe at very early ranks)

Trotman & Lalmas, SIGIR Poster 2006)

Retrieval tasks

- **Ad hoc retrieval:**
  "a simulation of how a library might be used and involves the searching of a static set of XML documents using a new set of topics"
  - CO topics
  - CAS (+S) topics
- **Several retrieval strategies**
  - Thorough retrieval
  - Focused retrieval
  - Relevant in context
  - Best in context

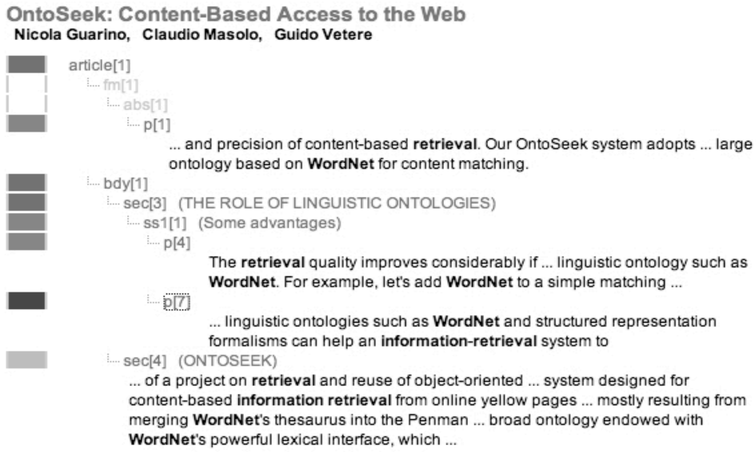
Presentation of results

Retrieval strategies

- Thorough strategy
  - core system's task underlying most XML retrieval strategies
  - estimate the relevance of all potentially retrievable elements in the collection
  - challenge is to rank elements appropriately, often for further processing
- Focused strategy:
  - find the most relevant and specific element on a path within a given document containing relevant information and return to the user only this most appropriate unit of retrieval
  - overlapping (redundant) results

Fetch & Browse

- Document ranking, and in each document, element ranking or set (called Relevant in Context in 2006)



(Courtesy of Sigurbjörnsson)

Best in context

- Document ranking, and in each document, return the best entry point
  - Element from where to start reading
  - Analysis:
    - Mostly not the beginning of the document
    - Often the element that is part of the first relevant fragment

(Kamp etal, SIGIR 2007 Poster)

CO retrieval task

- Specification:
  - make use of the CO topics
  - retrieves the most specific elements and only those, which are relevant to the topic
  - no structural constraints regarding the appropriate granularity
  - must identify the most appropriate XML elements to return to the user
- Two main strategies
  - Thorough strategy
  - Focused strategy

Thorough strategy

- Specification:
  - “core system's task underlying most XML retrieval strategies, which is to estimate the relevance of potentially retrievable elements in the collection”
  - overlap problem viewed as an interface and presentation issues
  - challenge is to rank elements appropriately
- Task that most XML approaches performed up to 2004 in INEX.

Focused strategy

- Specification:

  "find the most exhaustive and specific element on a path within a given document containing relevant information and return to the user only this most appropriate unit of retrieval"

  - no overlapping elements
  - return parent / child if same estimated relevance between parent and child elements
  - preference for specificity over exhaustivity

CAS retrieval task

- **Strict content-and-structure**:
  - retrieve relevant elements that exactly match the structure specified in the query (2002, 2003)

- **Vague content-and-structure**:
  - retrieve relevant elements that may not be the same as the target elements, but are structurally similar (2003)
  - retrieve relevant elements even if do not exactly meet the structural conditions; treat structure specification as hints as to where to look (since 2004)

CAS (+S) retrieval task

- Make use of CO+**S topics**: <castitle>
- **Structural hints**:
  - "Upon discovering that his/her <title> query returned many irrelevant elements, a user might decide to add structural hints, i.e. to write his/her initial CO query as a CAS query"

open standards for digital video in distance learning

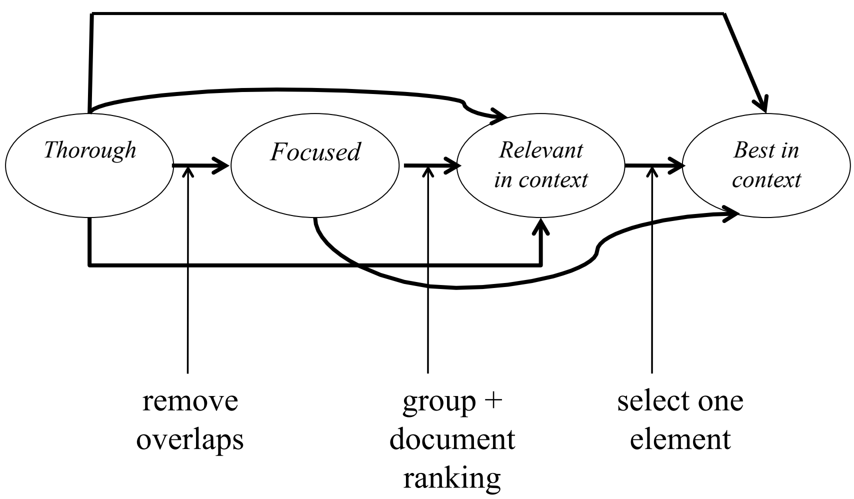
⇩

//article//sec[about(.,**open standards for digital video in distance learning**)]

- Two strategies (as for CO retrieval task):
  - *Focussed strategy*
  - *Thorough strategy*

(Trotman & Lalmas, SIGIR Poster 2006)

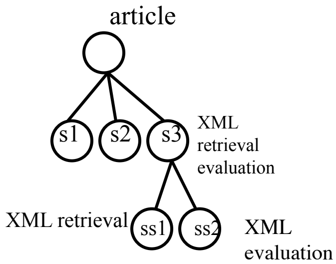
Relationships between the retrieval strategies



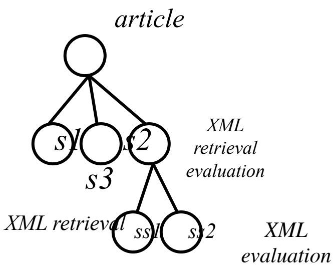
Relevance in XML retrieval

- A document is **relevant** if it "has significant and demonstrable bearing on the matter at hand".
- Common assumptions in laboratory experimentation:
  - **Objectivity**
  - **Topicality**
  - **Binary nature**
  - **Independence**

(Borlund, JASIST 2003)
(Goevert etal, JIR 2006)



Relevance in XML retrieval

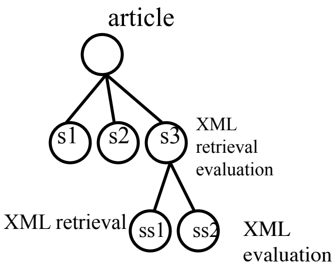


- **Topicality not enough**
- **Binary nature not enough**
- **Independence is wrong**

specificity: extent to which a document component is focused on the information need, while being an relevant unit.

(based on Chiaramella etal, FERMI fetch and browse model 1996)

Relevance in XML retrieval: INEX 2003 - 2004



- **Topicality not enough**
- **Binary nature not enough**
- **Independence is wrong**

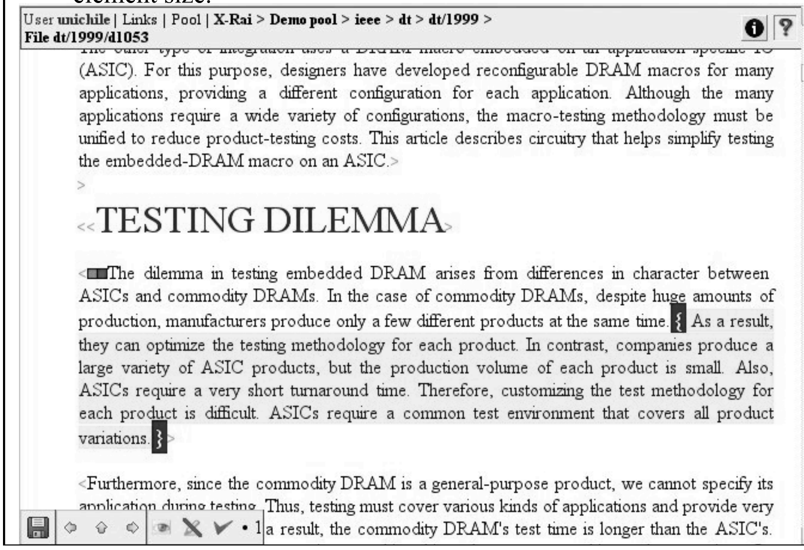
- **Relevance = (0,0) (1,1) (1,2) (1,3) (2,1) (2,2) (2,3) (3,1) (3,2) (3,3)**
  - **exhaustivity = how much the section discusses the query: 0, 1, 2, 3**
  - **specificity = how focused the section is on the query: 0, 1, 2, 3**
- **If a subsection is relevant so must be its enclosing section, ...**

Relevance - to recap

- find smallest component (→ specificity) that is highly relevant (→ exhaustivity)
- **specificity**: extent to which a document component is focused on the information need, while being an informative unit.
- **exhaustivity**: extent to which the information contained in a document component satisfies the information need.

Specificity dimension 2005 -

continuous scale defined as ratio (in characters) of the highlighted text to element size.



Exhaustivity dimension

Scale reduced to 3+1:

- Highly exhaustive (2): the element discussed most or all aspects of the query.
- Partly exhaustive (1): the element discussed only few aspects of the query.
- Not exhaustive (0): the element did not discuss the query.
- Too Small (?): the element contains relevant material but is too small to be relevant on it own.

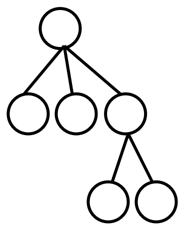
New assessment procedure led to better quality assessments
(Piwowarski etal, 2007)

Further simplification

- Statistical analysis on the INEX 2005 data:
  - The exhaustivity 3+1 scale is not needed in most scenarios to compare XML retrieval approaches
  - *The two small maybe simulated by some threshold length*
- INEX 2006-7 use only the specificity dimension to "measure" relevance
  - The same highlighting approach is used
  - *Some investigation being done regarding the two small elements*

(Ogilvie & Lalmas, 2006)

Relevance assessment task



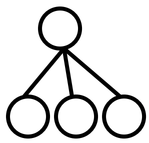
- Topics are assessed by the INEX participants
- Pooling technique (~500 elements on runs of 1500 elements)
- **Completeness**
  - **Rules that force assessors to assess related elements**
  - E.g. element assessed relevant → its parent element and children elements must also be assessed
  - …
- **Consistency**
  - **Rules to enforce consistent assessments**
  - E.g. Parent of a relevant element must also be relevant, although to a different extent
  - E.g. Exhaustivity increases going up; specificity increases going down
  - …

(Piwowarski & Lalmas, CIKM 2004)

Quality of assessments - Recap

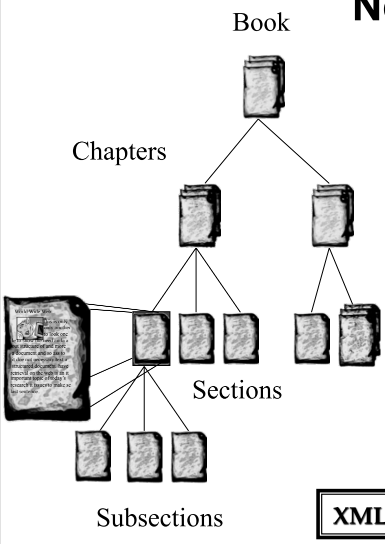
- Very laborious assessment task, eventually impacting on the quality of assessments (Trotman, Glasgow IR festival 2005)
  - binary document agreement is 27% (compared to TREC 6 (33%) and TREC 4 (42049%))
  - exact element agreement is 16%
- Interactive study shows that assessors agreement levels are high only at extreme ends of the relevance scale (very vs. not relevant) (Pehcevski *et al*, Glasgow IR festival 2005)
- Statistical analysis in 2004 data showed that comparisons of approaches would lead to same outcomes using a reduced scale (Ogilvie & Lalmas, 2006)
- A simplified assessment procedure based on highlighting (Clarke, Glasgow IR festival 2005)

Measuring effectiveness: Metrics



- Need to consider:
  - Multi-graded dimensions of relevance
  - Near-misses
- Metrics
  - inex_eval (also known as inex2002) (Goevert & Kazai, INEX 2002)
    official INEX metric 2002-2004
  - inex_eval_ng (also known as inex2003) (Goevert etal, JIR 2006)
  - ERR (expected ratio of relevant units) (Piwowarski & Gallinari, INEX 2003)
  - xCG (XML cumulative gain) (Kazai & Lalmas, TOIS 2006)
    official INEX metric 2005-2006
  - t2i (tolerance to irrelevance) (de Vries *et al*, RIAO 2004)
  - EPRUM (Expected Precision Recall with User Modelling) (Piwowarski & Dupret, SIGIR 2006)
  - HiXEval (Highlighting XML Retrieval Evaluation) (Pehcevski & Thom, INEX 2005)
    - Variant of it is now official INEX metric 2007- (Kamps *et al*, INEX 2007)
  - …

Near-misses

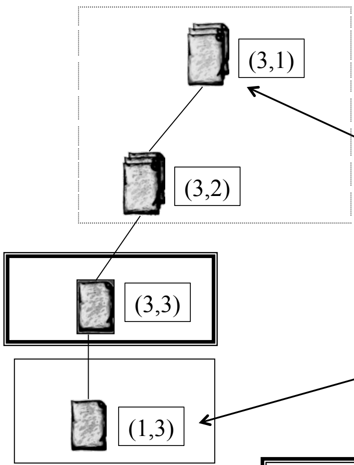


XML retrieval allows users to retrieve document components that are **more focused**, e.g. a section of a book instead of an entire book

BUT: what about if the chapter or one the subsections is returned?

XML SEARCHING = QUERYING + BROWSING

Near-misses (2004 scale)



XML retrieval allows users to retrieve document components that are **more focused**, e.g. a section of a book instead of an entire book

BUT: what about if the chapter or one the subsections is returned?

XML SEARCHING = QUERYING + BROWSING

(exhaustivity, specificity)

Retrieve the *best* XML elements according to content and structure criteria (2004 scale):

- Most exhaustive and the most specific = (**3**,**3**)

- Near misses = (**3**,**3**) + (2,**3**) (1,**3**) ← specific
- Near misses = (**3**, **3**) + (**3**,2) (**3**,1) ← exhaustive
- Near misses = (**3**, **3**) + (2,**3**) (1,**3**) (**3**,2) (**3**,1) (1,2) …

near-misses

Two multi-graded dimensions of relevance

- **Several "user models"**
  - **Expert and impatient**: only reward retrieval of highly exhaustive and specific elements (3,3) → *no near-misses*
  - Expert and patient: only reward retrieval of highly specific elements (3,3), (2,3) (1,3) → *(2,3) and (1,3) are near-misses*
  - …
  - **Naïve and has lots of time**: reward - to a different extent - the retrieval of any relevant elements; i.e. everything apart (0,0) → *everything apart (3,3) is a near-miss*

- **Use a quantisation function for each "user model"**

Examples of quantization functions

Expert and impatient

$$quant_{strict}(e,s) = \begin{cases} 1 & if\ (e,s) = (3,3) \\ 0 & otherwise \end{cases}$$

Naïve and has a lot of time

$$quant_{gen}(e,s) = \begin{cases} 1.00 & if \quad (e,s) = (3,3) \\ 0.75 & if \quad (e,s) \in \{(2,3),(3,2),(3,1)\} \\ 0.50 & if \quad (e,s) \in \{(1,3),(2,2),(2,1)\} \\ 0.25 & if \quad (e,s) \in \{(1,1),(1,2)\} \\ 0.00 & if \quad (e,s) = (0,0) \end{cases}$$

Using "standard" precision/recall

Simulated runs (Piwowarski & Gallinari, INEX 2003)

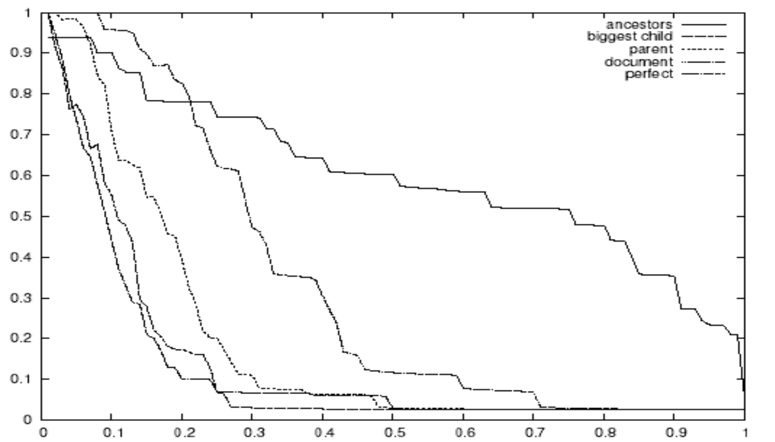


Figure 2. Generalised precision-recall. The axis of abscissas represents recall and the axis of ordinate the precision. Precision are averaged over the queries.

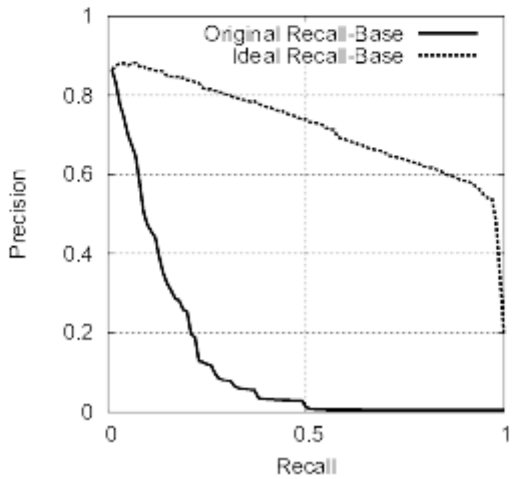
Overlap in results

Official INEX 2004 Results for CO topics

Rank	Systems (runs)	Avg Prec	Overlap
1.	IBM Haifa Research Lab (CO-0.5-LAREFIENMENT)	0.1437	**80.89**
2.	IBM Haifa Research Lab (CO-0.5)	0.1340	**81.46**
3.	University of Waterloo (Waterloo-Baseline)	0.1267	**76.32**
4.	University of Amsterdam (UAms-CO-T-FBack)	0.1174	**81.85**
5.	University of Waterloo (Waterloo-Expanded)	0.1173	**75.62**
6.	Queensland University of Technology (CO_PS_Stop50K)	0.1073	**75.89**
7.	Queensland University of Technology (CO_PS_099_049)	0.1072	**76.81**
8.	IBM Haifa Research Lab (CO-0.5-Clustering)	0.1043	**81.10**
9.	University of Amsterdam (UAms-CO-T)	0.1030	**71.96**
10.	LIP6 (simple)	0.0921	**64.29**

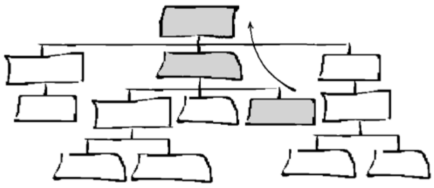
Overlap in recall-base

100% recall only if all relevant elements returned including overlapping elements



(Kazai etal, SIGIR 2004)

Relevance propagates up!



- **~26,000 relevant elements on ~14,000 relevant paths**
- **Propagated assessments: ~45%**
- **Increase in size of recall-base: ~182%**

(INEX 2004 data)

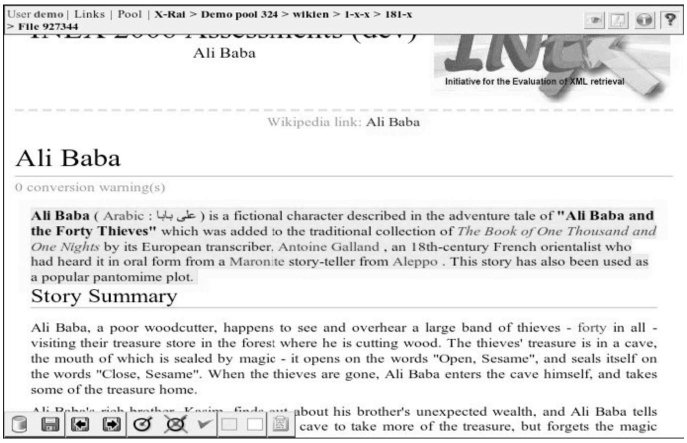
(Kazai etal, SIGIR 2004)

XCG: XML cumulated gain measures

- Based on cumulated gain measure for IR (Kekäläinen and Järvelin, TOIS 2002)
- Accumulate gain obtained by retrieving elements up to a given rank; thus not based on precision and recall → **user-oriented measures**
- Extended to include a precision/recall behaviour → **system-oriented measures**
- Require the construction of
  - an <u>ideal recall-base</u> to **separate what should be retrieved and what are near-misses**
  - an <u>associated ideal run</u>, which **contains what should be retrieved**
- with which retrieval runs are compared, which include what is being retrieved, **including near-misses**.

(Kazai & Lalmas, TOIS 2006)

HiXEval - Generalized precision and recall based on amount of highlighted content



For each element, we derive:
- rsize: number of highlighted characters
- size: number of characters

For each topic, we derive:
- Trel: number of highlighted characters in collection

HiXEval - Generalized precision and recall based on amount of highlighted content

- Precision at rank r

$$P(r) = \frac{\sum_{i=1}^{r} rsize(e_i)}{\sum_{i=1}^{r} size(e_i)}$$

- Recall at rank r

$$R(r) = \frac{1}{Trel} \cdot \sum_{i=1}^{r} rsize(e_i)$$

- F-measure at rank r, average precision, MAP, etc

(Pehcevski & Thom, INEX 2005; Kamps *et al*, INEX 2007)

Evaluation and INEX - Recap

- Larger and more realistic collection with Wikipedia
- Better understanding of information needs and retrieval scenarios
- Better understanding of how to measure effectiveness
  - Near-misses and overlaps
  - Application to other IR problems
- Who are the real users?
  - Larsen etal, SIGIR 2006 poster; Betsi etal, SIGIR 2006 poster; Pharo & Trotman, SIGIR Forum 2007.
- Book search track at INEX

Other INEX tracks

- Interactive (iTrack)
- Relevance feedback
- Natural language query processing
- Heterogeneous collection
- Multimedia track
- Document mining together with PASCAL network
- Use-case studies
- Entity ranking
- Link-the-Wiki
- Web service discovery
- Question Answering (QA@INEX)
- Efficiency
- Data-Centric

Conclusions

- Major advances in XML search (ranking) approaches made possible with INEX
- Evaluating XML retrieval effectiveness itself a research problem
- Many open problems for research

Areas for Open Problems

- DB and IR
  - Interaction between traditional DB query optimization (query rewriting) and ranking
- "Old" vs. new IR models
  - Combination of evidence problem
  - What evidence to use?
- Simple/succinct vs. complex/verbose QL
  - Define an XQuery core?

Areas for Open Problems

- Indexing & searching
  - Efficient algorithms
- INEX test collection and effectiveness
  - Too complex?
  - What constitutes a retrieval baseline?
  - Generalisation of the results on other data sets
- Quality evaluation (Web, XML)
  - Who are the users?
  - What are their information needs?
  - What are the requirements?

Outline

- Introduction to XML, basics and standards
- Document-oriented XML retrieval
- Evaluating XML retrieval effectiveness
- **Going beyond XML retrieval**

Beyond XML retrieval

- Focused retrieval
- Aggregated results
- Structural context summarization
- Beyond the logical structure

Focused retrieval

- Best performance obtained using evidence from element, document, and element size, and this whatever the model.
  - How can we apply this to other so-called “focused” retrieval problem?
  - What other evidence, e.g. semantic tags, should be used?
  - What combination formalism should be used?

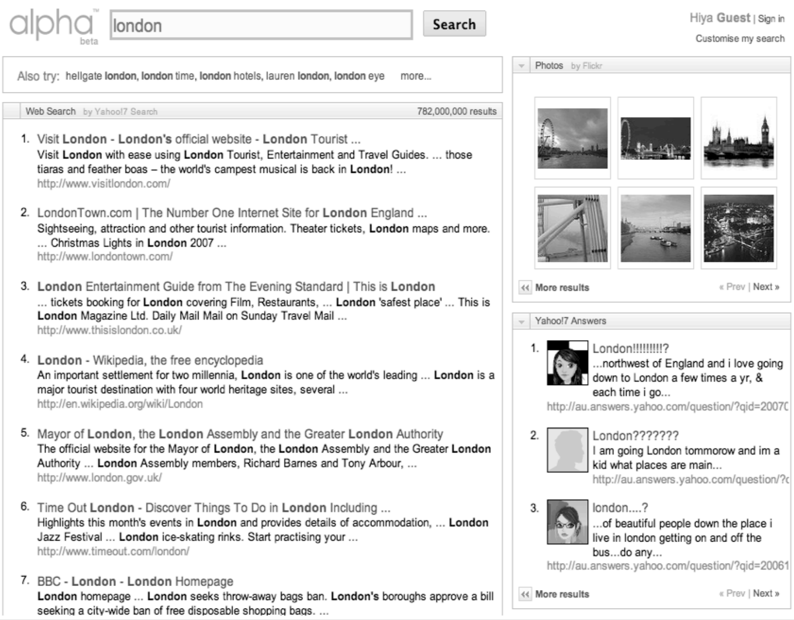
See workshop on focused retrieval at SIGIR 2007, SIGIR 2008

Aggregated results

- We know how to retrieve “snippets”.
- We know how to return “snippets” within a document (e.g. heatmap).
- How to combine/mix snippets from across documents to return **meaningful** aggregated results?
  - Virtual” documents (from Chiaramella)
  - This is not just a data fusion problem!
  - “User experience” in search engine

An example of aggregated search (1)

An example of aggregated search (2)



An example of aggregated search (3)

An example of aggregated search (4)

Figure 1: A dynamically generated SciFly PDF brochure

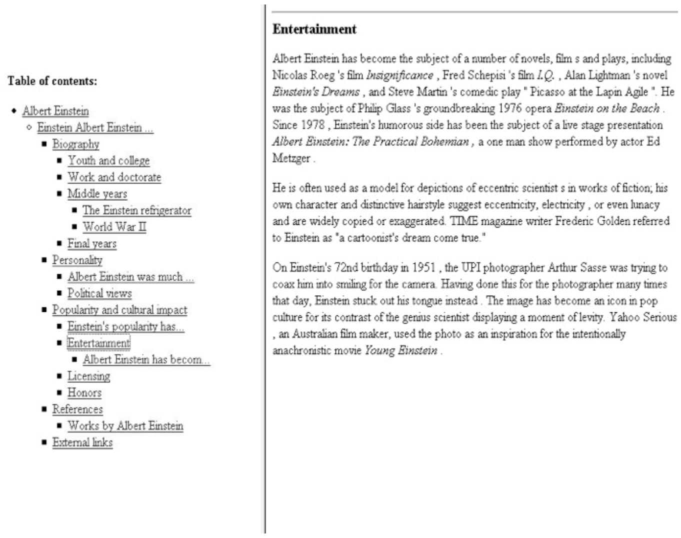
(Courtesy of Cecile Paris, CSIRO, Sydney, Australia)

Structural context summarization

- Users require document context when viewing an elements result
- We know how to summarize the structure (ToC) of a document (depth, relevance, etc)
- How can we summarize the structure of the search results, to provide context for the whole search.
  - Not just clusters
  - Refer to Amazon property list for a given product

XML retrieval systems display

Providing context for the element



Beyond the logical structure

- We know how to exploit the tags representing the logical structure to provide focused retrieval.
- What about other tags, e.g. semantic tags, formatting tags, template tags, etc?

Acknowledgements

- These slides are based on a number of presentations from the presenters at other events and from other researchers.
  - **Accessing XML content - An information retrieval perspective, Winter School, Champery, 2008**
  - **M. Consens, R. A. Baeza-Yates, M. Lalmas, S. Amer-Yahia: XML retrieval: DB/IR in theory, web in practice. VLDB 2007**
  - **S. Amer-Yahia, R. Baeza-Yates, M. Concens amd M. Lalmas. XML Retrieval: Integrated IR-DB Challenges and Solutions. SIGIR 2007.**
  - **S. Amer-Yahia and M. Lalmas. Accessing XML Content: From DB and IR Perspectives, CIKM 2005.**
  - R. Baeza-Yates and N. Fuhr. XML Retrieval, SIGIR 2004
  - R. Baeza-Yates and M. Consens. The Continued Saga of DB-IR Integration, SIGIR 2005
  - M. Lalmas. Structure/XML retrieval. ESSIR 2005, ESSIR 2007
  - M. de Rijke, J. Kamps and M. Marx. Retrieving Content and Structure, ESSLLI 2005
  - B. Sigurbjörnsson, Element Retrieval in Action, QMUL Seminar 2005
  - R. Baeza-Yates and M. Lalmas, XML Information Retrieval, SIGIR 2006

- S. Amer-Yahia & M. Lalmas. XML Search: Languages, INEX and Scoring, Submitted for Publication, 2006.
- E. Amitay, D. Carmel, R. Lempel & A. Soffer. Scaling IR-system evaluation using term relevance sets. SIGIR 2004, pp 10-17.
- P. Arvola, J. Kekäläinen & M. Junkkari. Query Evaluation with Structural Indices. INEX 2005.
- P. Arvola, M. Junkkari & J. Kekäläinen. Generalized contextualization method for XML information retrieval. CIKM 2005.
- R. A. Baeza-Yates, N. Fuhr & Y.S. Maarek. SIGIR XML and Information Retrieval workshop, SIGIR Forum, 36(2):53–57, 2002. 3.
- R. A. Baeza-Yates, Y. S. Maarek, T. Roelleke & A.P. de Vries. SIGIR joint XML & Information Retrieval and Integration of IR and DB workshops, SIGIR Forum, 38(2):24–30, 2004.
- R. Baeza-Yates, D. Carmel, Y.S. Maarek, and A. Sofer (eds). Special issue on XML Retrieval, JASIST, 53, 2002.
- R. Baeza-Yates & G. Navarro, Integrating contents and structure in text retrieval, SIGMOD 25:67-79, 1996.
- R. Baeza-Yates and G. Navarro, XQL and Proximal Nodes, JASIST 53:504-514, 2002.
- S. Betsi, M. Lalmas, A. Tombros & T. Tsikrika. User Expectations from XML Element Retrieval, SIGIR 2006 (Poster).

- Henk M. Blanken, T. Grabs, H.-J. Schek, R. Schenkel & G. Weikum (eds). Intelligent Search on XML Data, Applications, Languages, Models, Implementations, and Benchmarks, 2003.
- P. Borlund. The concept of relevance in IR. JASIS, 54(10):913-925, 2003.
- J.P. Callan. (1994). Passage-level evidence in document retrieval. SIGIR 1994.
- D. Carmel, Y.S. Maarek & A. Soffer. XML and Information Retrieval. SIGIR Forum, 34(1): 31–36, 2000.
- D. Carmel, Y.S. Maarek, M. Mandelbrod, Y. Mass & A. Soffer: Searching XML documents via XML fragments. SIGIR 2003.
- Y. Chiaramella, P. Mulhem & F. Fourel. A model for multimedia information retrieval. FERMI Technical report, University of Glasgow, 1996.
- Chinenyanga and Kushmerik, Expressive retrieval from XML documents, SIGIR 2001. 2001.
- C. Clarke. Range results in XML retrieval. INEX 2005 Workshop on Element Retrieval Methodology.
- C. Clarke. Controlling Overlap in Content-Oriented XML Retrieval. SIGIR 2005.
- W.S. Cooper. Expected search length: A single measure of retrieval effectiveness based on weak ordering action of retrieval systems. JASIS, 19:30-41, 1968.
- A. Delgado & R. Baeza-Yates. A Comparison of XML Query Languages, Upgrade 3, 12-25, 2002.
- L. Denoyer & P. Gallinari.The Wikipedia XML Corpus. SIGIR Forum, 40(1), 2006.

- A. de Vries, G. Kazai & M. Lalmas. Tolerance to Irrelevance: A User-effort Oriented Evaluation of Retrieval Systems without Predefined Retrieval Unit, RIAO 2004.
- N. Fuhr, N. Goevert, G. Kazai & M. Lalmas (eds). INitiative for the Evaluation of XML Retrieval (INEX 2002): Proceedings of the First INEX Workshop. ERCIM Workshop Proceedings, 2003.
- N. Fuhr & Kai Großjohann. XIRQL: A Query Language for Information Retrieval in XML Documents. SIGIR 2001.
- N. Fuhr, M. Lalmas & S. Malik (eds). INitiative for the Evaluation of XML Retrieval (INEX 2003). Proceedings of the Second INEX Workshop, 2004.
- N. Fuhr, M. Lalmas, S. Malik & G Kazai (eds). Advances in XML Information Retrieval and Evaluation: Fourth Workshop of the INitiative for the Evaluation of XML Retrieval (INEX 2005), LNCS 3977, 2006.
- N. Fuhr, M. Lalmas, S. Malik & Z. Szavik (eds). Advances in XML Information Retrieval, Third International Workshop of the Initiative for the Evaluation of XML Retrieval (INEX 2004), INEX 2004,LNCS 3493, 2005.
- S. Geva. GPX - Gardens Point XML IR at INEX 2005. INEX 2005.
- N. Goevert, N. Fuhr, M. Lalmas, & G. Kazai. Evaluating the effectiveness of content-oriented XML retrieval methods. Journal of Information Retrieval, 2006 (In Press).
- N. Goevert & G. Kazai. Overview of the INitiative for the Evaluation of XML retrieval (INEX) 2002, INEX 2002.

- M. A. Hearst & C. Plaunt. Subtopic structuring for full-length document access. SIGIR 1993.
- K. Järvelin & J. Kekäläinen. Cumulated gain-based evaluation of IR techniques. ACM TOIS 20(4):422–446, 2002.
- J. Kamps, M. de Rijke & B. Sigurbjornsson. Length normalization in XML retrieval. SIGIR 2004.
- J. Kamps, M. de Rijke & B. Sigurbjornsson. The importance of length normalization for XML retrieval. Information Retrieval, 8(4):631–654, 2005.
- G. Kazai & M. Lalmas. eXtended Cumulated Gain Measures for the Evaluation of Content-oriented XML Retrieval. ACM TOIS, 2006 (To appear).
- G. Kazai, M. Lalmas & A. de Vries. The overlap problem in content-oriented xml retrieval evaluation. SIGIR 2004.
- G. Kazai, M. Lalmas, N. Fuhr & N. Gövert. A report on the first year of the INitiative for the evaluation of XML retrieval (INEX 02). JASIST, 54, 2004.
- G. Kazai, M. Lalmas & J. Reid. Construction of a test collection for the focussed retrieval of structured documents. ECIR 2003.
- M. Lalmas & E. Moutogianni. A Dempster-Shafer indexing for the focussed retrieval of a hierarchically structured document space: Implementation and experiments on a web museum collection, RIAO 2000.
- M. Lalmas and I. Ruthven. Representing and Retrieving Structured Documents using the Dempster-Shafer Theory of Evidence: Modelling and Evaluation. JDoc, 54(5):529-565, 1998.

- B. Larsen, A. Tombros & S. Malik . Is XML retrieval meaningful to users? Searcher preferences for full documents vs. elements. SIGIR 2006 (Poster).
- Luk, Leong, Dillon,Chan, Croft & Allan, A Survey on Indexing and Searching XML, "Special Issue on XML and IR", *JASIST,* 2002.
- Mass, Mandelbrod, Amitay, and Soffer, JuruXML - an XML retrieval system at INEX 2002. INEX 2003.
- Y. Mass & M. Mandelbrod. Retrieving the most relevant XML Components. INEX 2004.
- Y. Mass & M. Mandelbrod. Using the INEX environment as a test bed for various user models for XML Retrieval. INEX 2005.
- V. Mihajlovic, G. Ramirez, T. Westerveld, D. Hiemstra, H. E. Blok & A. P. de Vries. TIJAH Scratches INEX 2005: Vague Element Selection, Image Search, Overlap, and Relevance Feedback. INEX 2005.
- Navarro and Baeza-Yates, Proximal Nodes, SIGIR 1995 (journal version in ACM TOIS, 1997).
- P. Ogilvie & M. Lalmas. Investigating the exhaustivity dimension in content-oriented XML element retrieval evaluation. 2006. Submitted for publication.
- Paul Ogilvie & Jamie Callan: Hierarchical Language Models for XML Component Retrieval. INEX 2004.
- Paul Ogilvie & Jamie Callan: Parameter Estimation for a Simple Hierarchical Generative Model for XML Retrieval. INEX 2005.
- J. Pehcevski & J. A. Thom. Hixeval: Highlighting xml retrieval evaluation. INEX 2005.

- J. Pehcevski, J.A. Thom & A.-M. Vercoustre. Hybrid XML Retrieval: Combining Information Retrieval and a Native XML Database. Journal of Information Retrieval 8(4): 571-600, 2005.
- J. Pehcevski, J. A. Thom & A.M. Vercoustre. Users and assessors in the context of INEX: Are relevance dimensions relevant? INEX 2005 Workshop on Element Retrieval Methodology.
- B. Piwowarski & G. Dupret. Evaluation in (XML) Information Retrieval: Expected Precision-Recall with User Modelling (EPRUM), SIGIR 2006.
- B. Piwowarski & P. Gallinari. Expected ratio of relevant units: A measure for structured information retrieval. INEX 2003.
- B. Piwowarski & M. Lalmas. Providing consistent and exhaustive relevance assessments for XML retrieval evaluation. CIKM 2004.
- B. Piwowarski, A. Trotman & M. Lalmas. Sound and complete relevance assessments for XML retrieval. 2006. Submitted for publication.
- V.V. Raghavan, P. Bollmann, & G. S. Jung. A critical investigation of recall and precision as measures of retrieval system performance. ACM TOIS, 7(3):205–229, 1989.
- G. Ramirez, T. Westerveld & A. P. de Vries. Using structural relationships for focused XML retrieval. FQAS 2006.
- T. Roelleke, M. Lalmas G. Kazai, I Ruthven & S. Quicker. The Accessibility Dimension for Structured Document Retrieval, ECIR 2002.
- G. Salton, J. Allan & C. Buckley. Approaches to Passage Retrieval in Full Text Information Systems, SIGIR 1993.

- K. Sauvagnat, L. Hlaoua & M. Boughanem XFIRM at INEX 2005: ad-hoc and relevance feedback tracks. INEX 2005.
- B. Sigurbjornsson, J. Kamps & M. de Rijke The Importance of Length Normalization for XML Retrieval. Journal of Information Retrieval,8(4), 2005.
- B. Sigurbjornsson, J. Kamps & M. de Rijke The Effect of Structured Queries and Selective Indexing on XML Retrieval. INEX 2005.
- B. Sigurbjornsson & A. Trotman. Queries: INEX 2003 working group report. INEX 2003.
- A. Trotman and M. Lalmas. Strict and Vague Interpretation of XML-Retrieval Queries, SIGIR 2006 (Poster).
- M. Theobald, R. Schenkel & G. Weikum. TopX & XXL at INEX 2005. INEX 2005.
- A. Tombros, S. Malik & B. Larsen. Report on the INEX 2004 interactive track. ACM SIGIR Forum, 39(1):43–49, 2005.
- A. Trotman. Wanted: Element retrieval users. INEX 2005 Workshop on Element Retrieval Methodology.
- A. Trotman & M. Lalmas. Why Structural Hints in Queries do not Help XML Retrieval, SIGIR 2006 (Poster).
- A. Trotman & B. Sigurbjornsson. NEXI, now and next. INEX 2004.
- A. Trotman & B. Sigurbjornsson. Narrowed extended XPATH I (NEXI). INEX 2004.
- C. J. van Rijsbergen. Information Retrieval. Butterworths, 1979.

- K. Sauvagnat, L. Hlaoua & M. Boughanem XFIRM at INEX 2005: ad-hoc and relevance feedback tracks. INEX 2005.
- B. Sigurbjornsson, J. Kamps & M. de Rijke The Importance of Length Normalization for XML Retrieval. Journal of Information Retrieval,8(4), 2005.
- B. Sigurbjornsson, J. Kamps & M. de Rijke The Effect of Structured Queries and Selective Indexing on XML Retrieval. INEX 2005.
- B. Sigurbjornsson & A. Trotman. Queries: INEX 2003 working group report. INEX 2003.
- A. Trotman and M. Lalmas. Strict and Vague Interpretation of XML-Retrieval Queries, SIGIR 2006 (Poster).
- M. Theobald, R. Schenkel & G. Weikum. TopX & XXL at INEX 2005. INEX 2005.
- A. Tombros, S. Malik & B. Larsen. Report on the INEX 2004 interactive track. ACM SIGIR Forum, 39(1):43–49, 2005.
- A. Trotman. Wanted: Element retrieval users. INEX 2005 Workshop on Element Retrieval Methodology.
- A. Trotman & M. Lalmas. Why Structural Hints in Queries do not Help XML Retrieval, SIGIR 2006 (Poster).
- A. Trotman & B. Sigurbjornsson. NEXI, now and next. INEX 2004.
- A. Trotman & B. Sigurbjornsson. Narrowed extended XPATH I (NEXI). INEX 2004.
- C. J. van Rijsbergen. Information Retrieval. Butterworths, 1979.

- J.-N. Vittaut & P. Gallinari. Machine Learning Ranking for Structured Information Retrieval. ECIR 2006.
- R. Wilkinson. Effective Retrieval of Structured Documents. SIGIR 1994.
- A. Woodley & S. Geva. NLPX at INEX 2004. INEX 2004.
- J.-N. Vittaut, B. Piwowarski & P. Gallinari. An Algebra for Structured Queries in Bayesian Networks. INEX 2004.