

# The Continued Saga of DB-IR Integration

Ricardo Baeza-Yates

[rbaeza@dcc.uchile.cl](mailto:rbaeza@dcc.uchile.cl)

[www.cwr.cl](http://www.cwr.cl)

*Center for Web Research  
Dept. of Computer Science  
Univ. of Chile*

Mariano P. Consens

[consens@mie.utoronto.ca](mailto:consens@mie.utoronto.ca)

[www.cs.toronto.edu/~consens](http://www.cs.toronto.edu/~consens)


*Information Engineering, MIE  
& Dept. of Computer Science  
University of Toronto*



## Agenda

1. Motivation
2. An Introduction to IR
3. Requirements for DB-IR
4. Semistructured Data
5. Industrial DB-IR Examples: Oracle, Verity
6. DB Approaches
7. IR & Hybrid Approaches
8. Open Problems
9. Bibliography





Very Large Data Bases

## Disclaimer

- This tutorial reflects the personal biases, preferences and limitations of the presenters ☺
- It does not cover everything done in DB+IR

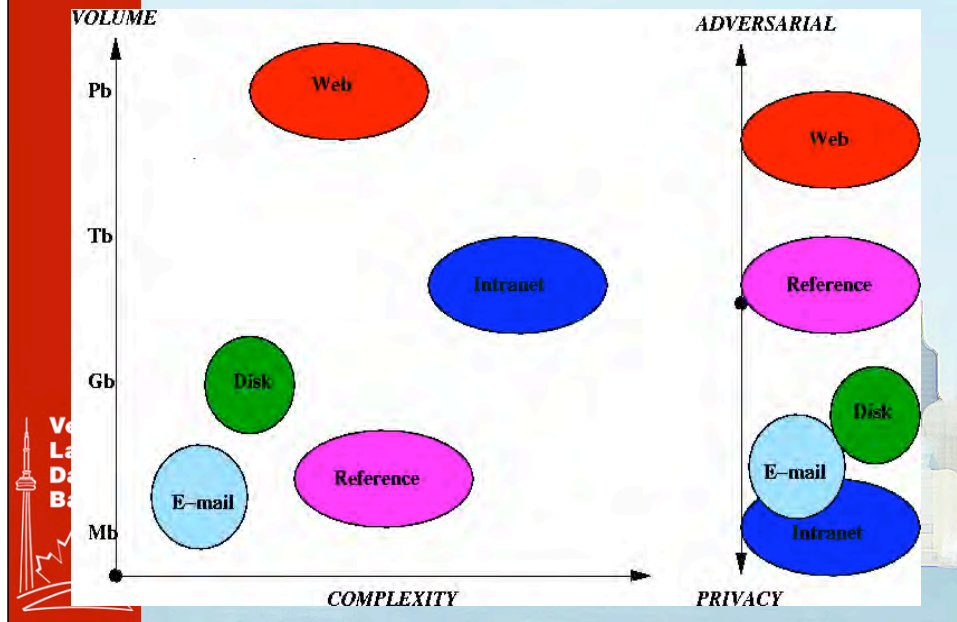


Very Large Data Bases

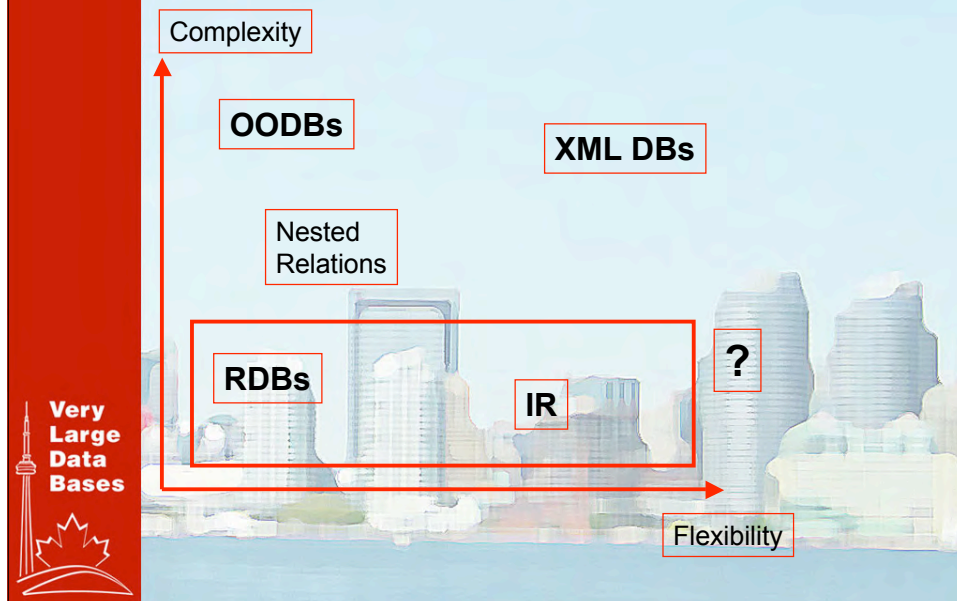
## 1. Motivation

- Types of data
- DB & IR Views
- Possible Solutions
- Applications
- Search Problems

## Different Views on Data



## Data and Databases

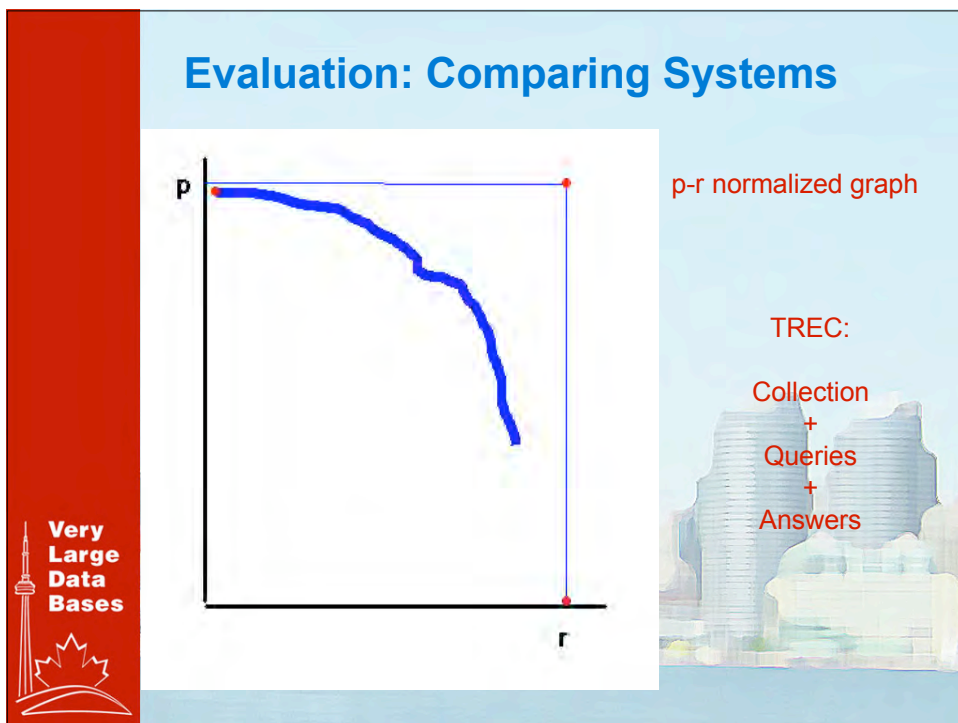
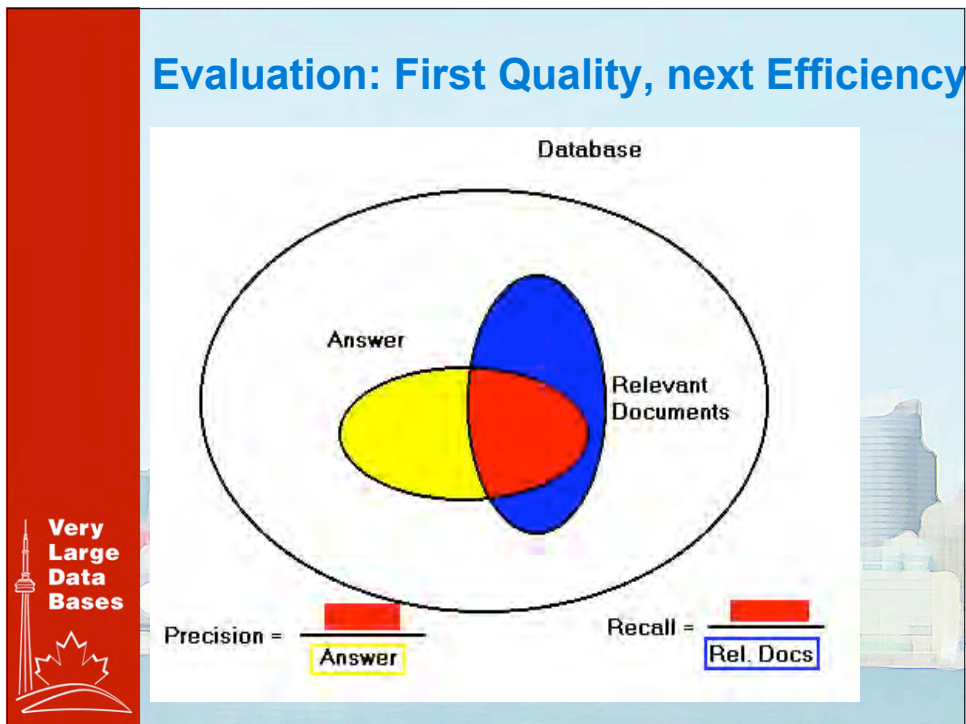


## RDB vs. IR

- 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
- 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

## 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**





## 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

## 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 Data Base?
- Are RDBs really a special case of IR systems?

## Applications for Integrated Systems

- E-commerce search
- Intranets & enterprise data
- Customer support (e.g. CRM)
- News archives, bulletin boards, etc.
- Personal information (e.g. My Life Bits)
- P2P Web Search



## Challenges posed by the Web

- Integration of autonomous data sources
  - Data/information integration
- Supporting heterogeneous data
  - How to do effective querying in the presence of structured and text data
  - How to support IR-style querying on DBs
    - Because now users seem to know IR/keyword style querying more, even though structure is good because it supports structured querying!
  - How to support imprecise queries





## Enterprise Search is Different

- Sophisticated systems run by librarians are morphing into simple self-service web-based search
  - Must be scalable, reliable, highly available
- Data is different
  - Heterogeneous in format & structure (documents, DBs, etc)
  - Less volume & better quality
- Searching is also different
  - Less & better queries, different tasks
  - Focus in recall rather than precision
- Other issues: security, able to search but not to see



## What is a Bad Interface/Result?

- No search box
- Inability to judge user intent
  - No spell checking
  - No context disambiguation (cricket: game or bug?)
  - No recommendation system, no user feedback
- Too many hits: answer overload
  - Return 10,000 hits when the average user looks only at the top-20
- The most relevant item is not at the top of the list
- Too many similar documents
  - Poor duplicate detection, poor clustering/categorization
- Inability to understand why a document has been returned
  - No KWIC
- Lack of Meta information
  - Size, format, date, etc.



## Cost of a Bad Search

- Information is useless if no one can find it
  - ROI for employee productivity
  - ROI for customer satisfaction
  - Cost of people using out-of-date information
  - Cost of people using wrong information
  - Cost of recreating information which cannot be found
  - Cost of opportunity for not finding the information



## Some Examples - I

Where is the search box?



## Some Examples – II

“ultra seek” or “ultraseek”?

The screenshot shows the Verity search engine interface. At the top, there are navigation links: Home, Company, Customers, Products, Partners, and a search bar. Below the navigation bar, the search results are displayed. The current query is 'ultra seek', and there are 62 documents found. The results are listed in a table with columns for Date, Doc Type, Title & summary, and Score. The first four results are from 2003/09/08, 2003/07/23, 2003/07/23, and 2003/07/15, all with a score of 72% or 70%.

Date	Doc Type	Title & summary	Score
2003/09/08	Verity, Inc. : Company : Events : Webinars	Summary: By Giga Information Group Senior Analyst Laura Ramos More and more companies are taking advantage of taxonomies with content management systems, portals and search technology to retrieve results organized for context and user needs. Ms. Feldman.... Folder: archive Size: 24KB	72%
2003/07/23	Verity, Inc. : Partner Program : Portlet Program : Epicentric	Summary: Epicentric, Inc. is a leading provider of Business Portal solutions for global 2000 companies to deliver integrated Web services to their customers, partners and employees. Contact Veri.... Folder: epicentric Size: 16KB	72%
2003/07/23	Verity, Inc. : Company : Corporate : Verity Awards : Industry Analysts	Summary: The Gartner Group: Verity listed as a leader in the Magic Quadrant for Enterprise Search Source: The Gartner Group: 2002 Enterprise Search Magic Quadrant, Feb., 2002 Verity was determined by Gartner to be listed as a ?Leader? in Enterprise Search. ID..... Folder: awards Size: 17KB	72%
2003/07/15	Verity, Inc. 2002 Annual Report & 10-K	Summary: Verity K2 Enterprise Verity K2 Enterprise provides the integrated three-tier foundation of next generation business portals and e-business applications. Verity K2 Developer Verity K2 Developer is a toolkit that lets commercial software de.....	70%



## Some Examples - III

Looking for “k-means” in lotus.com

The screenshot shows the IBM search engine interface. At the top, there are navigation links: Home, Products & services, Support & downloads, My account, and a search bar. Below the navigation bar, the search results are displayed. The current query is 'k-means', and there are 3 results found. The results are listed in a table with columns for Title & summary, and Score. The first three results are from 2003/09/08, 2003/07/23, and 2003/07/15, all with a score of 72% or 70%.

Date	Doc Type	Title & summary	Score
2003/09/08	Verity, Inc. : Company : Events : Webinars	Summary: By Giga Information Group Senior Analyst Laura Ramos More and more companies are taking advantage of taxonomies with content management systems, portals and search technology to retrieve results organized for context and user needs. Ms. Feldman.... Folder: archive Size: 24KB	72%
2003/07/23	Verity, Inc. : Partner Program : Portlet Program : Epicentric	Summary: Epicentric, Inc. is a leading provider of Business Portal solutions for global 2000 companies to deliver integrated Web services to their customers, partners and employees. Contact Veri.... Folder: epicentric Size: 16KB	72%
2003/07/23	Verity, Inc. : Company : Corporate : Verity Awards : Industry Analysts	Summary: The Gartner Group: Verity listed as a leader in the Magic Quadrant for Enterprise Search Source: The Gartner Group: 2002 Enterprise Search Magic Quadrant, Feb., 2002 Verity was determined by Gartner to be listed as a ?Leader? in Enterprise Search. ID..... Folder: awards Size: 17KB	72%
2003/07/15	Verity, Inc. 2002 Annual Report & 10-K	Summary: Verity K2 Enterprise Verity K2 Enterprise provides the integrated three-tier foundation of next generation business portals and e-business applications. Verity K2 Developer Verity K2 Developer is a toolkit that lets commercial software de.....	70%

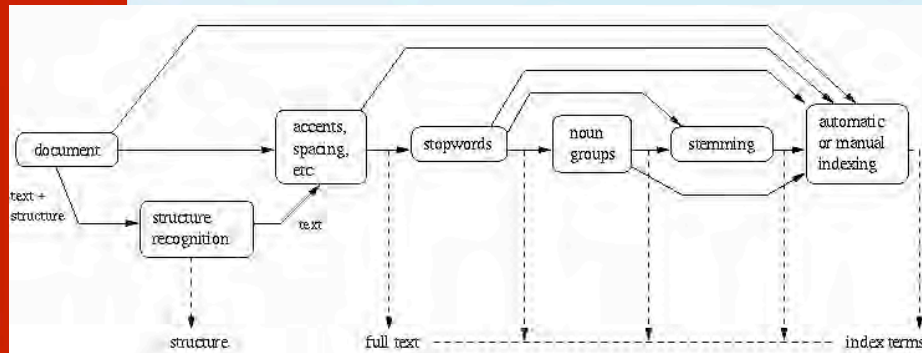


## 2. Introduction to IR through Web Retrieval

- IR challenges posed by the Web
- Logical view of text
- Similarity models
- IR system architecture
- IR query languages & interfaces

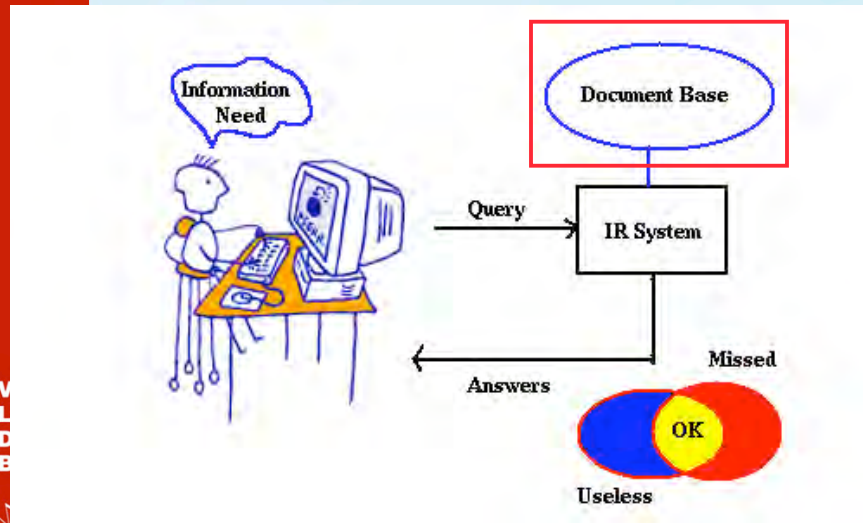


## Bag-of-Words Representation



Full-text continuum:  
ambiguity vs. completeness trade-off

## Challenges in Current IR Systems



## Document Base: Web

- Largest public repository of *data* (more than 6 billion static pages?)
- Today, there are more than 60 million Web servers
- Well connected graph with out-link and in-link power law distributions



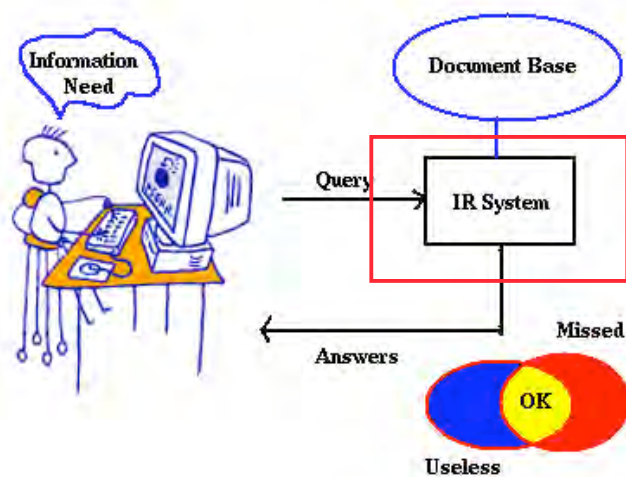


## Web Retrieval

- Problems:
  - volume
  - fast rate of change and growth
  - dynamic content
  - redundancy
  - organization and data quality
  - diversity
  - .....
- Deal with data overload

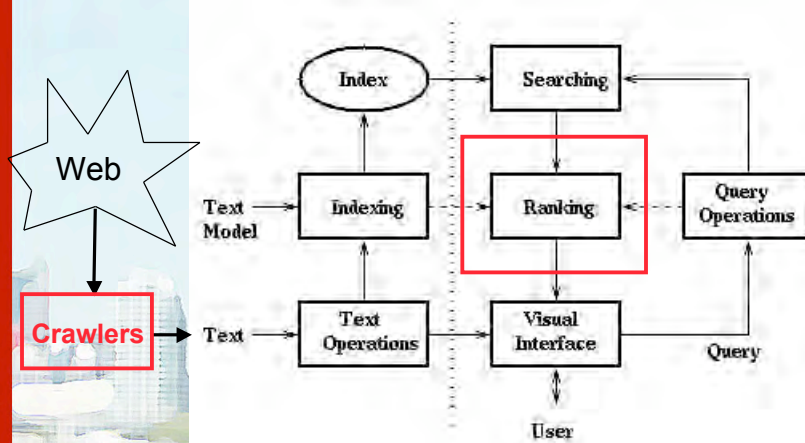


## Challenges in Current IR Systems



## Web Retrieval Architecture

Centralized parallel architecture



## Algorithmic Challenges

- Crawling:
  - Quantity
  - Freshness
  - Quality
  - Politeness vs. Usage of Resources
- Ranking
  - Words, links, usage logs, ... , metadata
  - Spamming of all kinds of data
  - Good precision, unknown recall

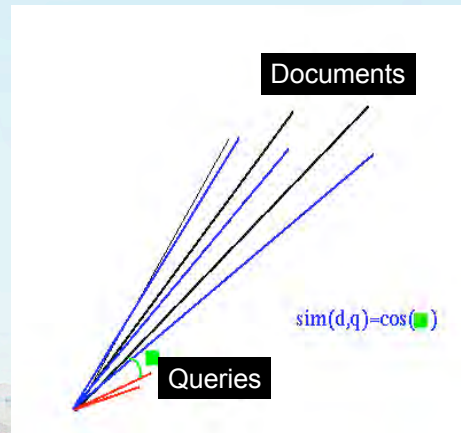


## Text Similarity Models

### Vector model:

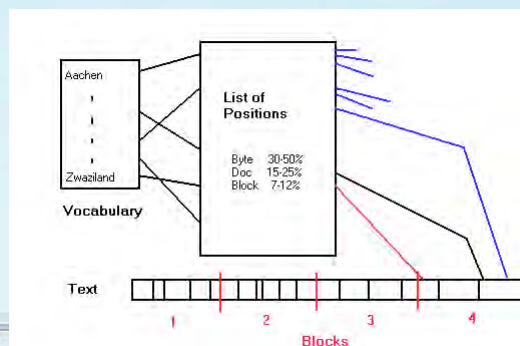
- words are dimensions
- *tf-idf* is used for weights

- Set Models:
  - Boolean, Fuzzy sets, ...
- Algebraic Models:
  - Vector, LSI, etc.
- Probabilistic Models:
  - Probabilistic, Inference & belief networks

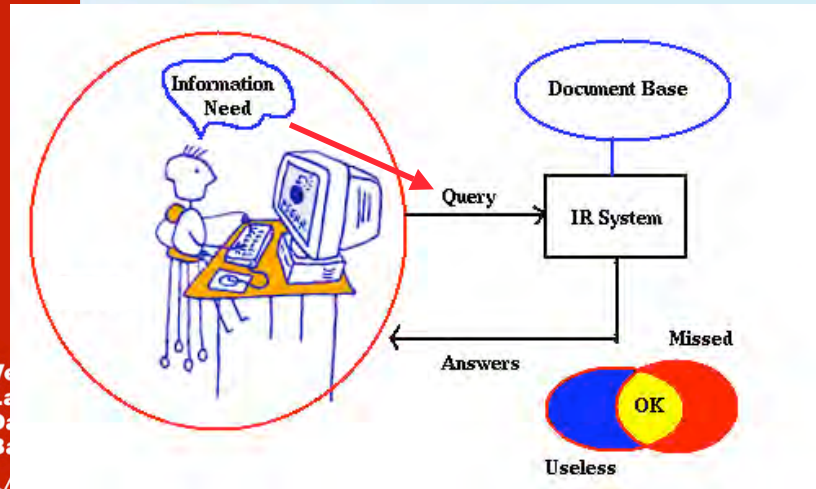


## Index

- Inverted index
- Lists sorted by weight
  - global (e.g. Pagerank)
  - local (e.g. word weights)
- Hashing + set operations
- Incremental updates



## Challenges in Current IR Systems

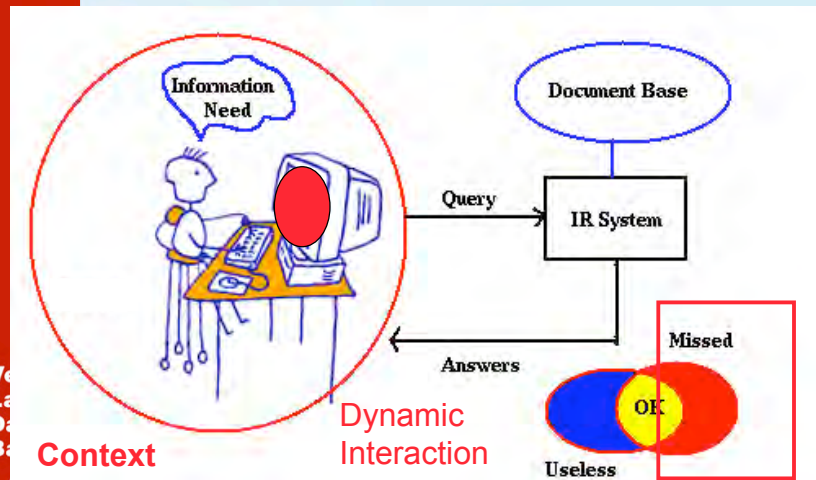


## Web Users

- Cultural and educational diversity
- Short queries
  - Inherent to users or due to the query language?
- Different goals:
  - Information need
  - Navigational need
  - Transactional need
- Short patience
  - few queries posed & few answers seen
- Other problems: concurrency, scale, ...

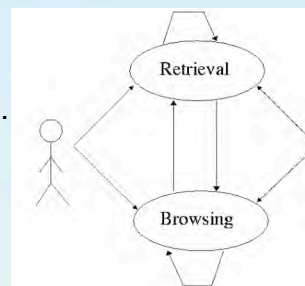


## Challenges in Current IR Systems



## Interaction

- Inexperienced users
- Dynamic information needs
- Varying task: querying, browsing, .
- No content overview
- Poor query language, no help
- Poor preview, no visualization
- Missing answers: partial Web coverage, invisible Web, different words or media, ...
- Useless answers






Very  
Large  
Data  
Bases

### 3. Requirements for DB-IR

- Motivating Applications
- Data and Query Requirements
- Sample Use Cases



Very  
Large  
Data  
Bases

### Sample Paper on the Web

#### XQL and Proximal Nodes

Ricardo Baeza-Yates · Gonzalo Navarro

Depto. de Ciencias de la Computación  
Universidad de Chile  
Blanco Encalada 2120  
Santiago 6511224, Chile  
E-mail: {rbaeza,gnavarro}@dcc.uchile.cl

#### Abstract

We consider the recently proposed XQL language, which is designed to query XML documents by content and structure. We show that an already existing model, namely “Proximal Nodes”, is the only one that addresses all the complex querying operations defined by XQL and that suggests an efficient implementation for them.

#### 1. Introduction

Searching on structured text is becoming more important with the increased use of XML. Although SGML existed for a long time, its complexity was the main limitation for a wider use. By taking advantage of the structure, content queries can be made more precise. Also, XML data can be seen as the meeting point between the database community (in particular the work on semi-structured data and query languages for XML) with the information retrieval community (structured text models). Our main goal in this paper is to show the

## Bibliography Entry

```
<proceedings>
  <inproceedings>
    <author>Ricardo Baeza-Yates</author>
    <author>Gonzalo Navarro</author>
    <title>XQL and Proximal Nodes</title>
    ...
  </inproceedings>
</proceedings>
```

- Describes metadata for the workshop article
- The XML data conforms to the DBPL schema (DTD)

## Paper Content in XML

```
<workshop date="28 July 2000">
  <title>XML and Information Retrieval: A SIGIR 2000 Workshop</title>
  <editors>David Camel, 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 ...
      </section>
      ...
      <cite xmlns:xlink="http://www.acm.org/sigir/.../paper/xmlql">...</cite>
    </paper>
    ...
  </workshop>
```

- The XML data conforms to the publisher's DTD



**Very  
Large  
Data  
Bases**

## A Digital Library Application

- Web interface for the citation

**Access Content**



**Citations**

**Similar Documents**




**Very  
Large  
Data  
Bases**

## Applications Areas

- Scientific, Technical and Medical Reference Books, Journals, Publications
- Case Law and Litigation Materials
- Regulatory and Business Filings
- Maintenance, Repairs and Operations Manuals
- Product Documentation
  - Design
  - Procurement (SRM)
  - Customer Service (CRM)
- Collaboration
- Web, Intranet, Group & Personal Repositories





Very  
Large  
Data  
Bases


## Data Requirements

- Text, Documents, Images, Application Files, Multimedia Content
- Structured Data
  - Relations: Refers (From, To)
  - Hierarchies: proceedings/paper/section
- Semi-structured Data
  - Editorial comments on the paper

Assumption: XML data provides a reasonable most general case

Objects

Nested



Very  
Large  
Data  
Bases

## Publishing Relational Data

USERS

USERID	NAME	RATING
--------	------	--------

ITEMS

ITEMNO	DESCRIPTION	OFFERED_BY	RESERVE_PRICE
--------	-------------	------------	---------------

BIDS

USERID	ITEMNO	BID_AMOUNT	BID_DATE
--------	--------	------------	----------

```


<users>
  <user_tuple>
    <userid>
      1243
    </userid>
    <name>
      humphrey
    </name>
    <rating>
      ...
        
```

```

<items>
  <item_tuple>
    <itemno>
      1066
    </itemno>
    <description>
      unicycle
    </description>
    <offered_by>
      ...
        
```

```

<bids>
  <bid_tuple>
    <userid>
      1243
    </userid>
    <itemno>
      1066
    </itemno>
    <bid_amount>
      ...
        
```



Very Large Data Bases


## Queries on Views - Integration

```
<users>
  <user_tuple>
    <userid>
      1243
    </userid>
    <name>
      humphrey
    </name>
    <rating>
      ...
  ...
```

```
<items>
  <item_tuple>
    <itemno>
      1066
    </itemno>
    <description>
      unicycle
    </description>
    <offered_by>
      ...
  ...
```

```
<bids>
  <bid_tuple>
    <userid>
      1243
    </userid>
    <itemno>
      1066
    </itemno>
    <bid_amount>
      ...
  ...
```

```
<bidlisting>
  <bid>
    <user>
      <userid> 1243 </userid> <name> humphrey </name>
      <rating> ...
    </user>
    <item>
      <itemno> 1066 </itemno> <descr> unicycle </descr>
      <offered_by> ...
    </item>
    <bid_amount>
      ...
  ...
```



Very Large Data Bases

## Heterogeneous Sources - P2P

```
<users>
  <user_tuple>
    <userid>
      1243
    </userid>
    <name>
      humphrey
    </name>
    <rating>
      ...
  ...
```

```
<items>
  <item_tuple>
    <itemno>
      1066
    </itemno>
    <description>
      unicycle
    </description>
    <offered_by>
      ...
  ...
```

```
<bids>
  <bid_tuple>
    <userid>
      1243
    </userid>
    <itemno>
      1066
    </itemno>
    <bid_amount>
      ...
  ...
```

```
<bidlisting>
  <bid>
    <user>
      <userid> 1243 </userid> <name> humphrey </name>
      <rating> ...
    </user>
    <item>
      <itemno> 1066 </itemno> <descr> unicycle </descr>
      <offered_by> ...
    </item>
    <bid_amount>
      ...
  ...
```



**Very Large Data Bases**

## Query Requirements Overview

- Developing the web application





**Very Large Data Bases**

## Basic Query Requirements

- Express arbitrary Full-Text (FT) searches
- Select the substructures where the FT condition applies (*search context*)
- Select the substructures to be returned (*return context*)
- Choose how to determine relevance (and the context that applies)
- Access and combine the relevance scores
- Limit answer to top-k
- Full composition of FT and structural queries

23

## Additional Requirements

- Efficient and scalable query evaluation, supported by
  - Indexes (FT and structural)
  - Optimizer
- Rich functionality for presenting answers
  - Visual interfaces
  - Highlight the FT terms *in context*
- Support integration scenarios
- Query heterogeneous structure
  - Single collection
  - Crawled repository
  - Peer Sources

## Sample Use Cases

- Quick overview of the range of possible DB-IR requirements
  - Identify search and return contexts
  - Motivate relevance
  - Illustrate composition
- Taken from Full-text XQuery  
([//www.w3.org/TR/xmlquery-full-text-use-cases](http://www.w3.org/TR/xmlquery-full-text-use-cases))




## Finding Text in Elements

- Find all book titles containing the word "usability"
- Find all books with the phrase "usability tests" in book or chapter titles
  - Multiple search contexts, different return
- Find all books with the phrase "usability tests" (even across elements)
- Find all book titles for books with abstracts mentioning software developers (interpreted as having broad terms "software" near "developer")
  - Proximity
  - Thesaurus (developer, programmer)

## Finding Text in Structure

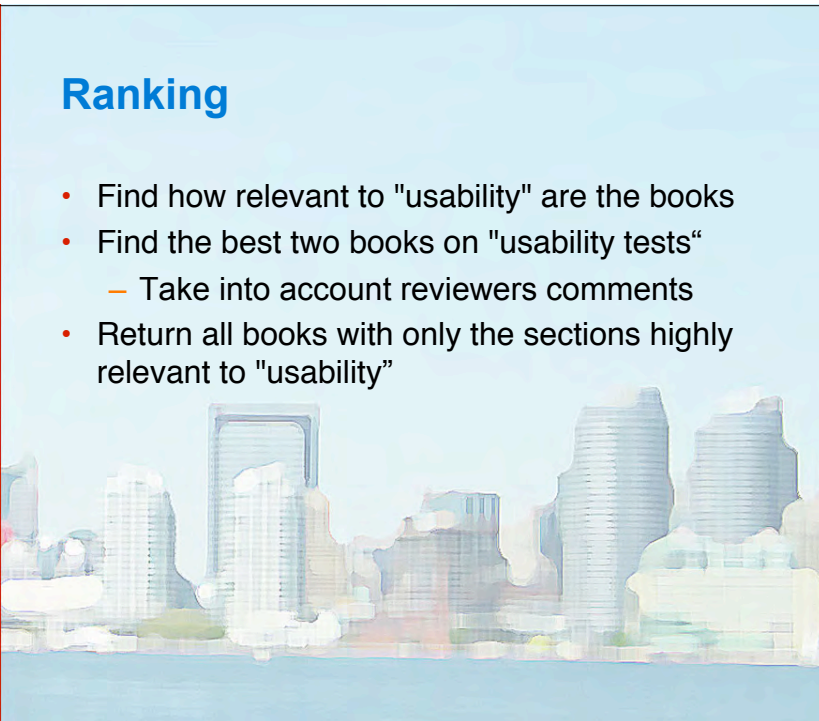
- Find the first two sections mentioning "task" in chapters on "conducting usability tests" with the book abstract not mentioning "software"
  - Structured search contexts
    - book/chapter//section
    - book/chapter
    - book/abstract
- Do the above ignoring footnotes in chapters but not in abstracts
  - Modifies the search contexts




Very  
Large  
Data  
Bases

## Ranking

- Find how relevant to "usability" are the books
- Find the best two books on "usability tests"
  - Take into account reviewers comments
- Return all books with only the sections highly relevant to "usability"

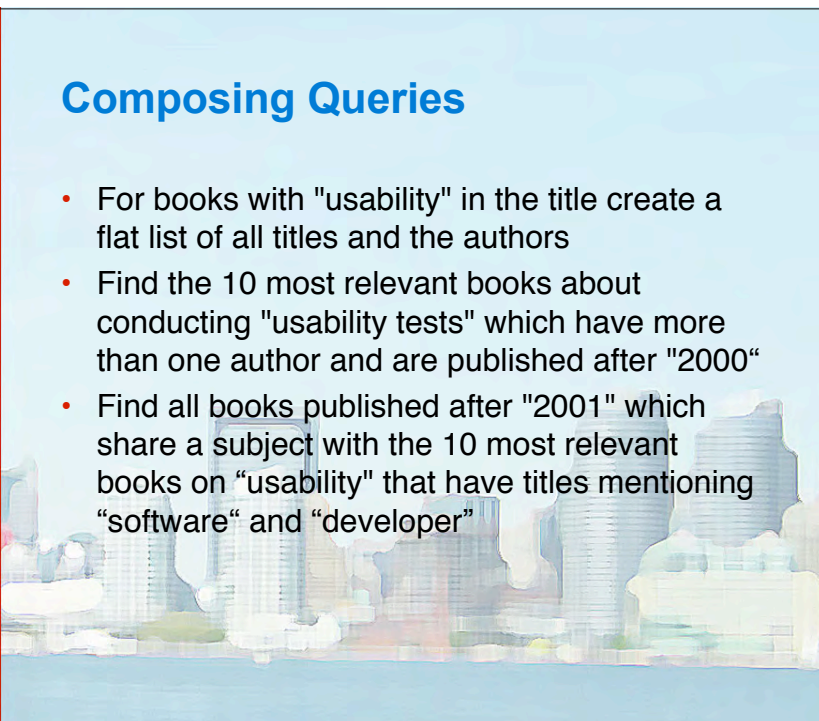




Very  
Large  
Data  
Bases

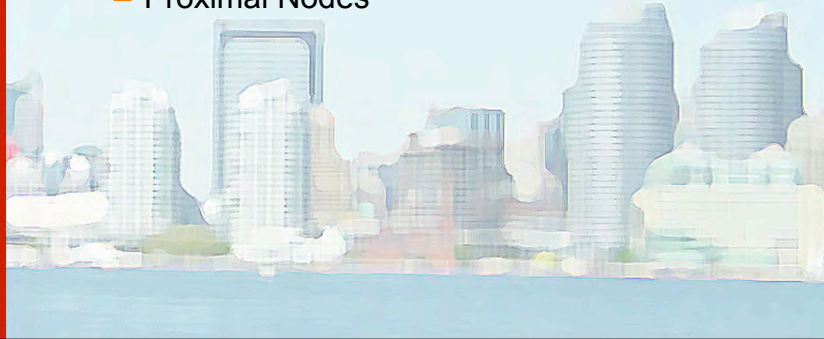
## Composing Queries

- For books with "usability" in the title create a flat list of all titles and the authors
- Find the 10 most relevant books about conducting "usability tests" which have more than one author and are published after "2000"
- Find all books published after "2001" which share a subject with the 10 most relevant books on "usability" that have titles mentioning "software" and "developer"



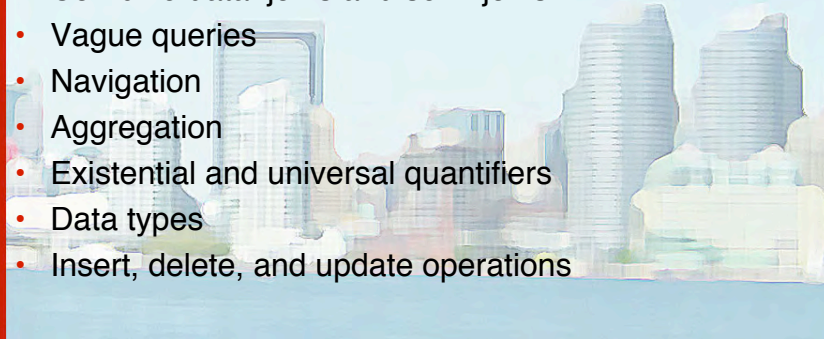
## 4. Semistructured Data

- Requirements for an XML Query Language
- XQuery
- XQuery & Full-text
- Structured Text Models
  - Proximal Nodes



## XML Requirements from DBs

- Selection: pattern + filter + constructor
- Filtering
- Reduction: pruned elements
- Restructuring: grouping, sorting, etc.
- Combine data: joins and semi-joins
- Vague queries
- Navigation
- Aggregation
- Existential and universal quantifiers
- Data types
- Insert, delete, and update operations

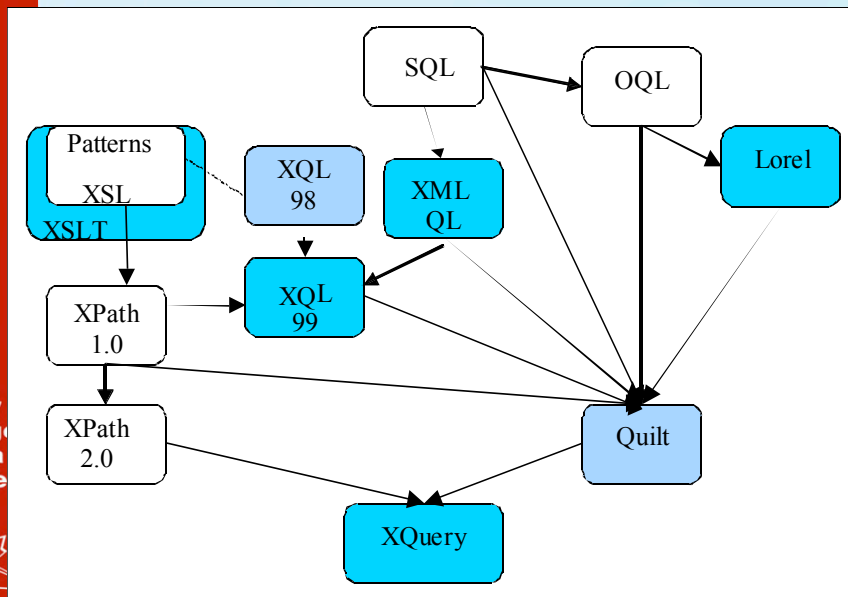


## XML Requirements from IR & Others

- Keyword queries: Boolean, context, similarity, etc.
- Pattern matching
- Structural queries: inclusion, distance relations, etc.
- Weighted query terms
- Ranking (Scoring)
- Use of metadata
- DTD or Xscheme awareness
- Support for Xlink and Xpointer
- Set operations on results

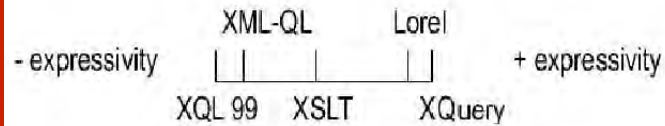


## XQuery History





## XML Query Language Comparison



	Lorel	XSLT	XML-QL	XQL 99	XQuery
<b>Main functions</b>	Queries of semi-structured data	Transformation of documents	Data queries, transformations, integration of XML data from different sources	Queries within a document and queries on collections of documents	Queries on heterogeneous data sources
<b>Data model</b>	Graph / Tree	Tree (such as XPath 1.0)	Graph	Tree (DOM of XML)	Ordered sequence of nodes (such as XPath 2.0)
<b>Input source &amp; format</b>	XML Documents	XML Document/s + StyleSheet	XML Documents from different sources	XML Document/s	XML Document, XML Fragments, Collections of XML documents
<b>Output information</b>	XML Document (Ordered list of identifiers of the resulting elements)	XML Document (Transformed XML tree), Collections of XML documents (xsl:document)	XML Document (XML Fragments)	XML Document (XML Fragments, List of resulting elements)	XML Document, XML Fragment, Collections of XML documents

## XML Query Language Comparison

		Lorel	XSLT	XML-QL	XQL 99	XQuery
Selection Operation	Pattern/ Filter/ Constructor	select constructor from pattern where filter	<xsl:for-each select= pattern > <xsl:if match=filter> <copy-of /> </xsl:if> </xsl:for-each>	WHERE pattern IN source, filter CONSTRUCT constructor	pattern [filter]	FOR patterns LET bindings WHERE filter RETURN constructor
	Relational Operators	>, >=, <, <=, =, <> ==	>, >=, <, <=, =, !=	>, >=, <, <=, =, !=	>, >=, <, <=, =, !=	>, >=, <, <=, =, != For nodes: ==, !=
	Boolean Operators	and, or, not	and, or	No	and, or	AND, OR
	Nesting queries	Yes	Yes	Yes	Yes	Yes
	Creation of new elements	Yes	Yes	Yes	No	Yes
Filtering of elements preserving hierarchy		No	Yes (using templates)	No	Yes	Yes (filter)
Reduction		No	Yes	No	Yes	No
Restructuring operations	Grouping of results	Yes (group by)	No	No	Only by structure, not by value	Yes
	Skolem Functions	Yes	No	Yes	No	Yes
	Sorting of results	Yes (order by)	Partial (xsl:sort <sup>3</sup> )	Yes (ORDER-BY)	No	Yes (SORTBY)
Inter-document links (join), Intra-documents links (semi-join)		Join, Semi-join	Semi-join	Join, semi-join	Semi-join, join	Join, semi-join

## XML Query Language Comparison

	Lorel	XSLT	XML-QL	XQL 99	XQuery
Use of tag variables	Yes	Yes	Yes	No	Yes
Path expressions	Regular expression operators: *,  , +, ? Qualifiers: >, @	XPath Expressions	Regular expression operators *,  , +, ?	Wild card: * Path Operators: /, //	XPath Expressions
Dereferencing of IDREF(S) attributes	Yes (As a subelement using the point notation)	Yes (id())	Yes (By means of a join)	Yes (id())	Yes (Dereference Operator =>)
Set Functions	min, max, count, sum, avg	sum, count	min, max, count, sum, avg	sum, count	min, max, count, sum, avg
Quantifiers					
Existential	Yes (exists)	Yes (implicit)	Yes (implicit)	Yes (implicit)	Yes (SOME)
Universal	Yes (for all)	No	No	Yes (all)	Yes (EVERY)
Handling of datatypes (XML Schema)	Partial	No (under study)	No	No	Yes
Insertion, delete and update	Yes	Yes	No	No	No



## XML Query Language Comparison

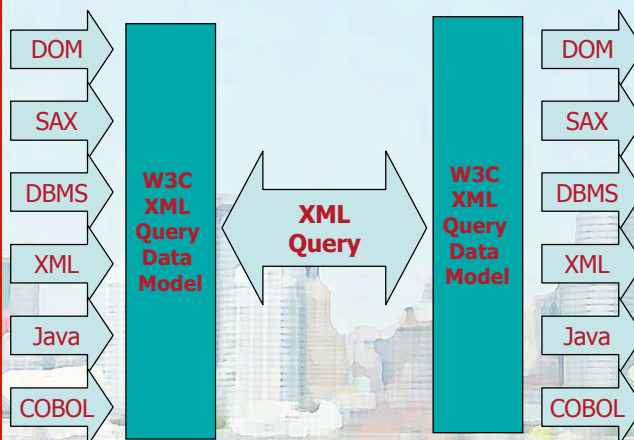
	Lorel	XSLT	XML-QL	XQL 99	XQuery
Keywords	A word inside free text	By means of path expressions	By means of path expressions	By means of path expressions	By means of path expressions
	Similarity	No	No	No	No
	Context	No	No	No	No
	Boolean Operators	Yes	Yes	No	Yes
Pattern matching		operators: like, grep, soundex	String operators and functions	Like operator	String operators and functions
Structural Queries	Structural Inclusion	By means of path expressions	By means of path expressions	By means of path expressions	By means of path expressions
	Positional Inclusion	Yes	Yes	Yes	Yes
	Structural proximity	No	No	No	Yes (immediately precedes *, *)
	Structural Order	By means of comparison of positional indexes	Yes (preceding, preceding-siblings, following, following-siblings)	By means of comparison of positional indexes	Yes (before, after)
Assignment of weighting to the terms of the query		No	No	No	No
RDF support		No	No	No	No
XLink and Xpointer support		No	No	No	Partial
Operations over sets		Intersection, union, difference	Union, difference	Intersection, union	Intersection, union, difference


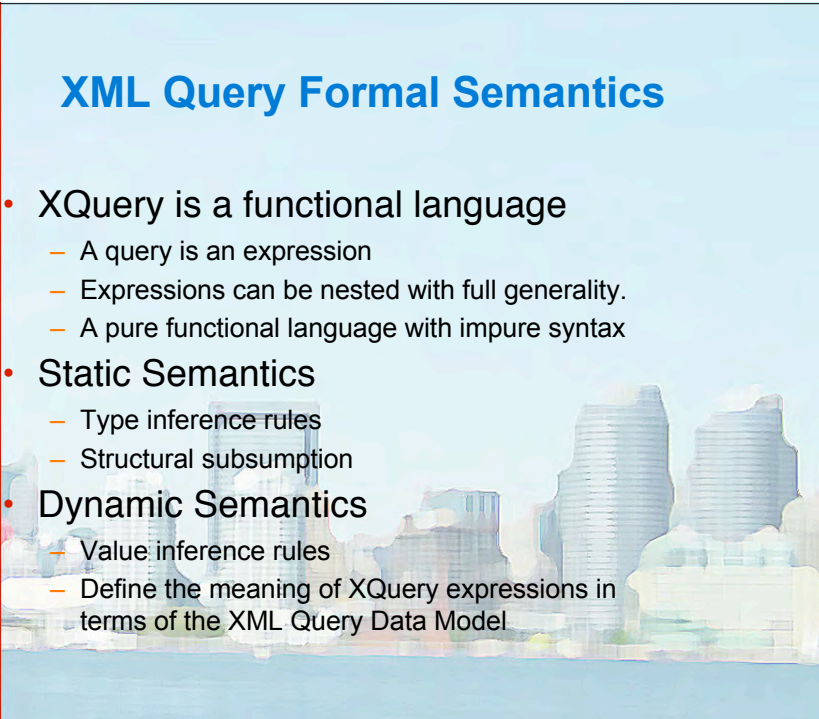


## XML Query Data Model

- Joint with XPath 2.0, XSL 2.0
  - Last version of Feb 2004
- Ordered, labeled forest
- Based on XML Information Set, PSVI
- Has node identity
- DTDs (from SGML, IR style)
- XML Scheme (DB style)
  - Provide data types


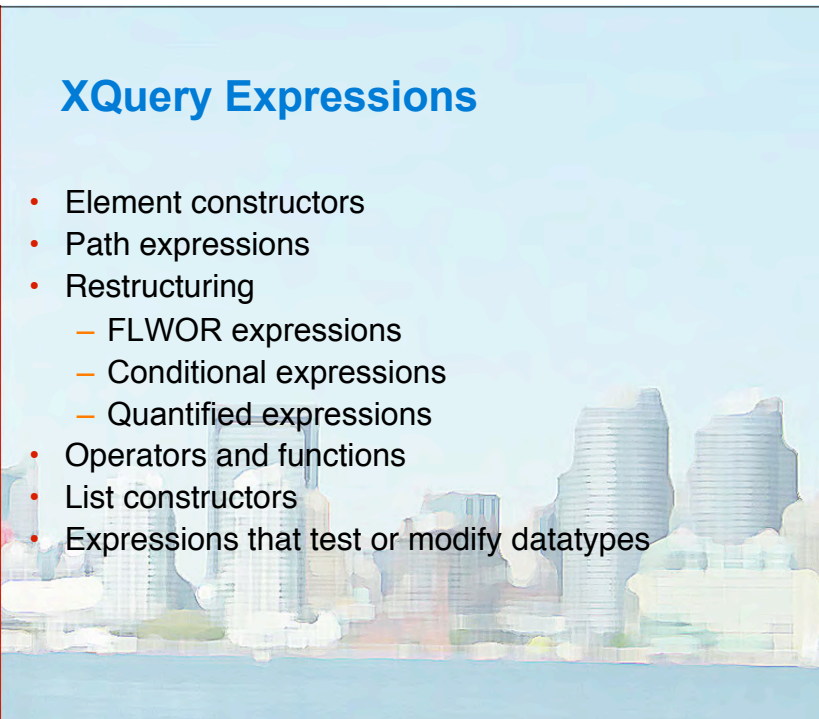
## XQuery and the Data Model





## XML Query Formal Semantics

- XQuery is a functional language
  - A query is an expression
  - Expressions can be nested with full generality.
  - A pure functional language with impure syntax
- Static Semantics
  - Type inference rules
  - Structural subsumption
- Dynamic Semantics
  - Value inference rules
  - Define the meaning of XQuery expressions in terms of the XML Query Data Model



## XQuery Expressions

- Element constructors
- Path expressions
- Restructuring
  - FLWOR expressions
  - Conditional expressions
  - Quantified expressions
- Operators and functions
- List constructors
- Expressions that test or modify datatypes



## Path Expressions

<bib>

<book year="1994">

<title>TCP/

<author>

<last>Stev

<first>W.

</author>

<publisher>

<price> 65.

</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

## FLWOR Expressions

- FOR - LET - WHERE - ORDER BY - RETURN
- Similar to SQL's SELECT - FROM - WHERE

for \$book in document("bib.xml")//book  
where \$book/publisher = "Addison-Wesley"  
return

<book>

{  
\$book/title,  
\$book/author  
}

</book>

## 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(.)
```

## Inner Join

"List names of users and descriptions of the items they offer"

- SQL:

```
SELECT u.name, i.description
FROM users AS u, items AS i
WHERE u.userid = i.offered_by
ORDER BY name, description;
```

- XQuery:

```
FOR $u IN //user_tuple, $i IN //item_tuple
WHERE $u/userid = $i/offered_by
RETURN
  <offering> {
    $u/name,
    $i/description
  } </offering> ORDERBY(name, description)
```



## Text Search

```
<section><title>Procedure</title>
```

The patient was taken to the operating room where she was placed in a supine position.

```
<anesthesia>
```

```
</anesthesia>
```

```
<prep> <action>
```

```
bladder</action>
```

```
and the abdominal
```

```
</prep>
```

```
<incision>A c
```

```
<geography>
```

```
</geography>
```

and the subcutaneous tissue was divided

```
<instrument>using electrocautery.</instrument>
```

```
</incision>
```

### Conditions on Text

Equality:

```
//section[title="Procedure"]
```

Full-text:

```
//section[contains(title, "Procedure")]
```

## Full-text Requirements - I

- Full-text predicates and SCORE functions are independent
- Full-text predicates use a language subset of SCORE functions
- Allow the user to return and sort-by SCORE (0..1)
- SCORE must not require explicit global corpus statistics
- SCORE algorithm should be provided and can be disabled
- Problems:
  - Not clear how to rank without global measures
  - Many/no answers problems
  - Search then rank is not practical
  - How to integrate other SCORE functions?





## Full-text Requirements - II


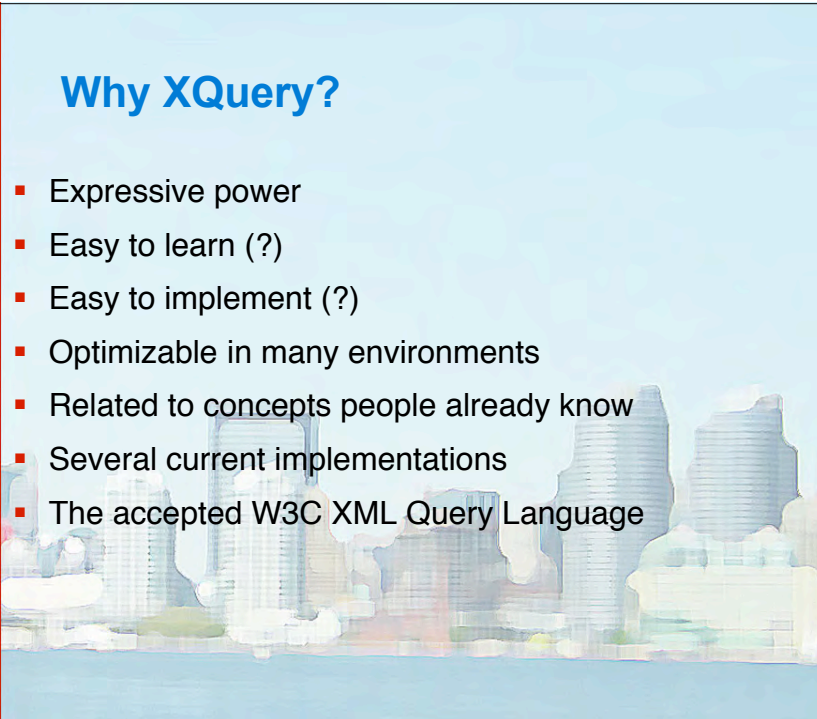
- Minimal operations:
  - Single-word and phrase search with stopwords
  - Suffix, prefix, infix
  - Proximity searching (with order)
  - Boolean operations
  - Word normalization, diacritics
  - Ranking relevance
- Search over everything, including attributes
- Proximity across markup elements
- Extensible



## XQuery Implementations


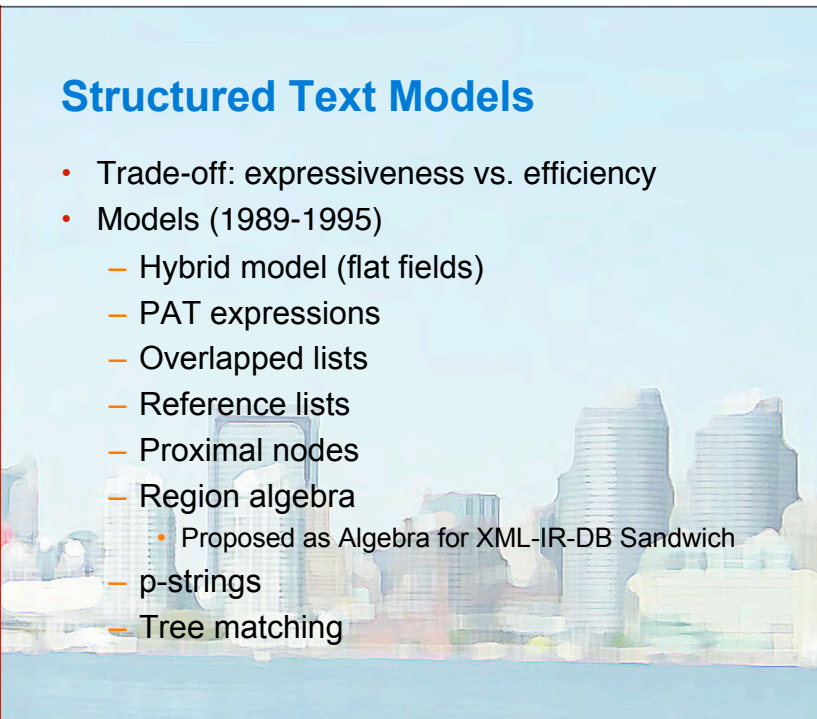
- Software AG's Tamino XML Query
- Microsoft, Oracle,
- Lucent Galax
- GMD-IPSNitem X-Hive
- XML Global
- SourceForge XQuench, Saxon, eXist, XQuery Lite
- Fatdog
- Qexo (GNU Kawa) - compiles to Java byte code
- Openlink, CL-XML (Common Lisp), Kweelt,...
- Soda3, DB4XML and about 15 more





## Why XQuery?

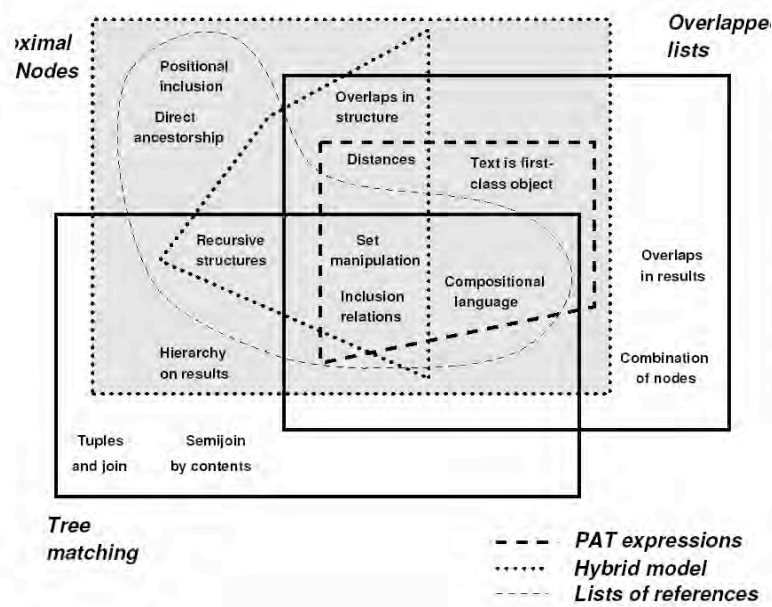
- Expressive power
- Easy to learn (?)
- Easy to implement (?)
- Optimizable in many environments
- Related to concepts people already know
- Several current implementations
- The accepted W3C XML Query Language



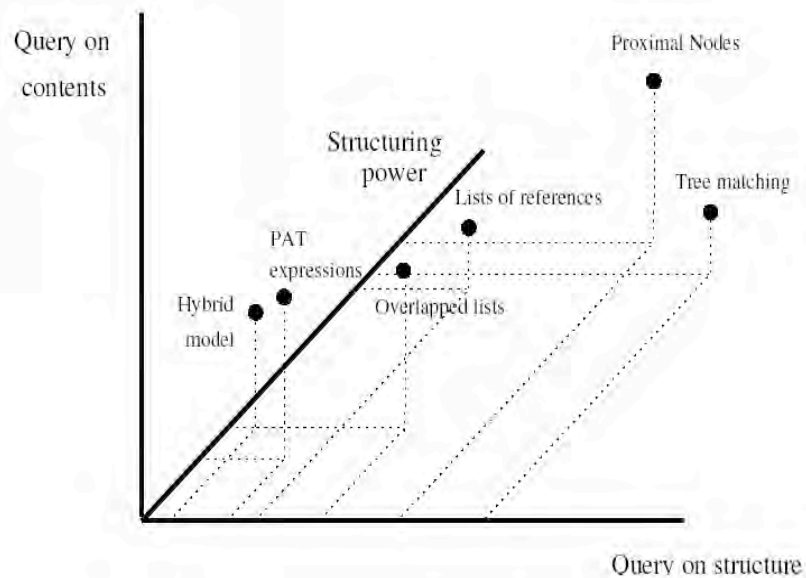
## Structured Text Models

- 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

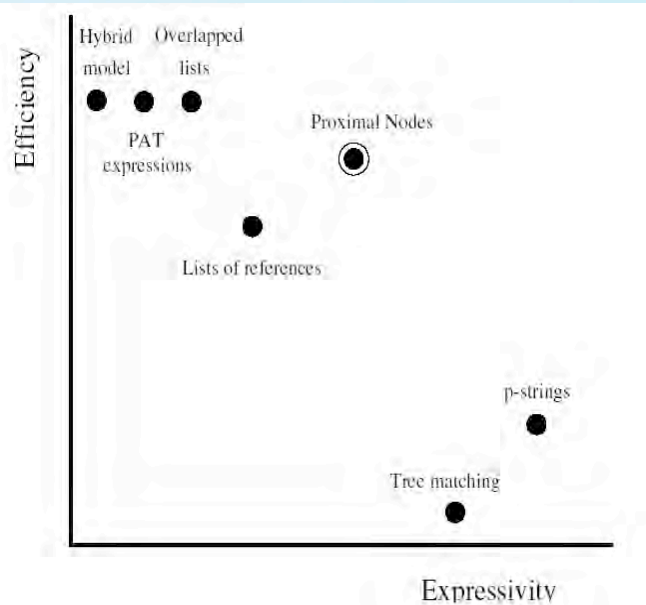
## Comparison - I



## Comparison - II



## Comparison - III



## Example: Proximal Nodes (Navarro & Baeza-Yates)

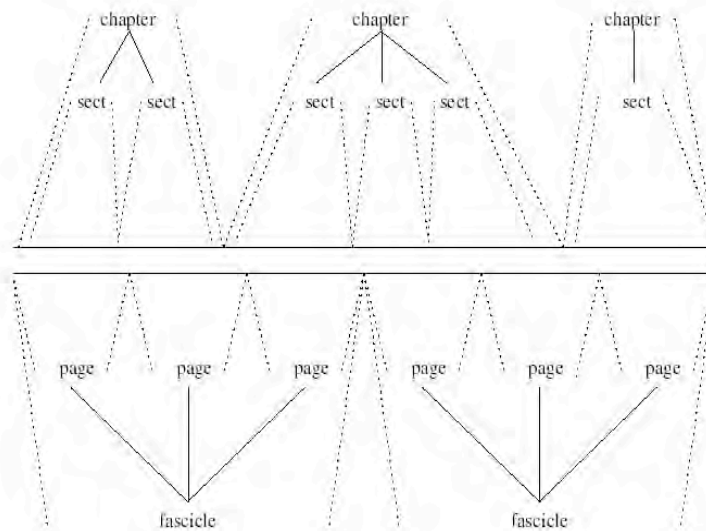
- Hierarchical structure
- Set-oriented language
- Avoid traversing the whole database
- Bottom-up strategy
- Solve leaves with indexes
- Operators work with near-by nodes
- Operators cannot use the text contents
- Most Xpath and Xquery expressions can be solved using this model



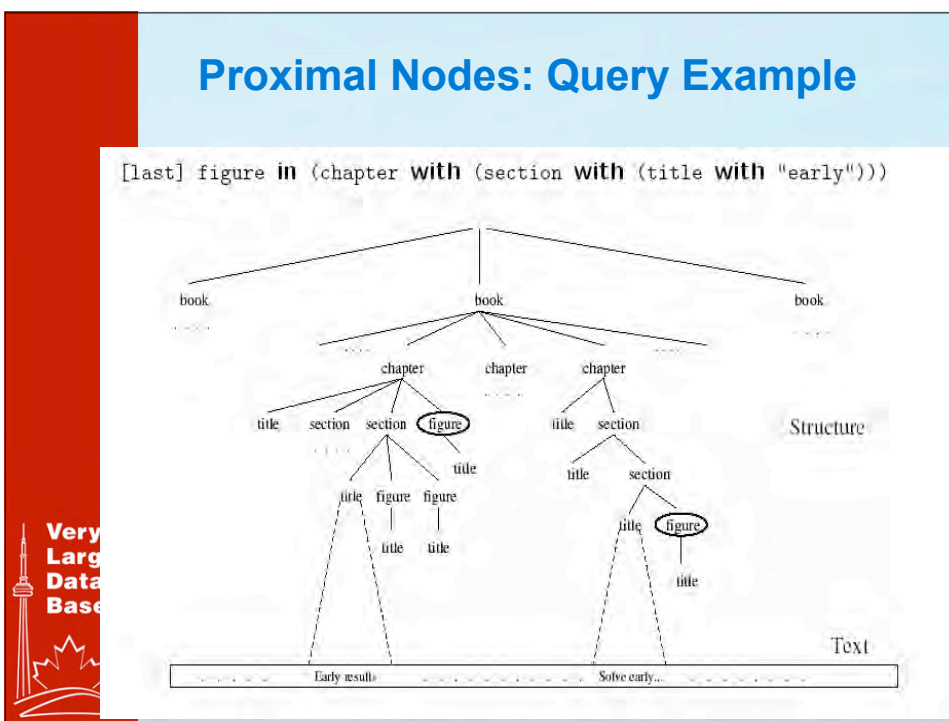
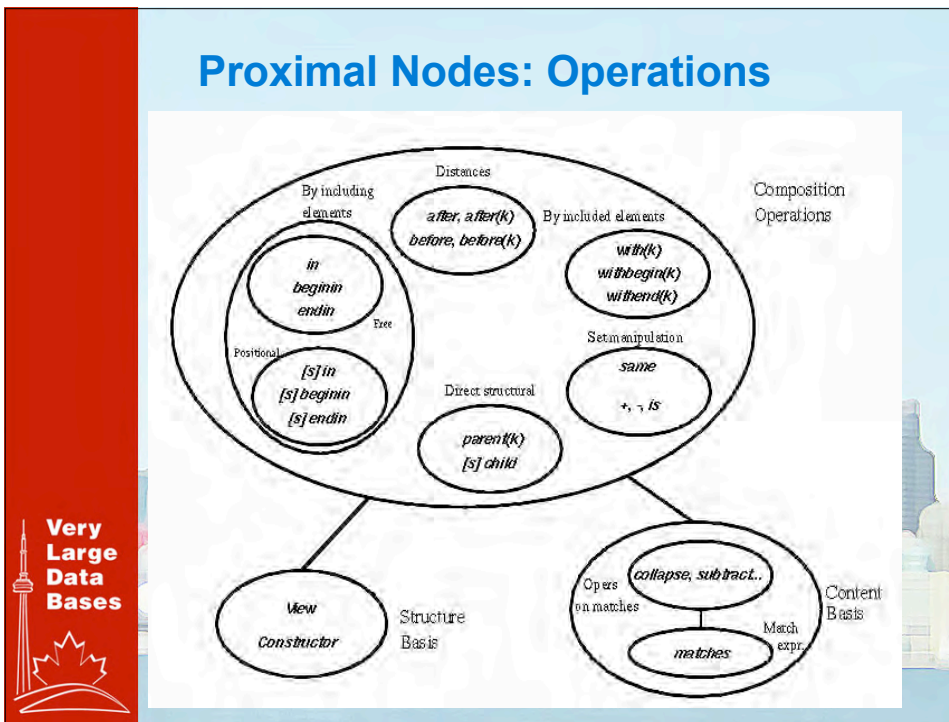
## Proximal Nodes: Data Model

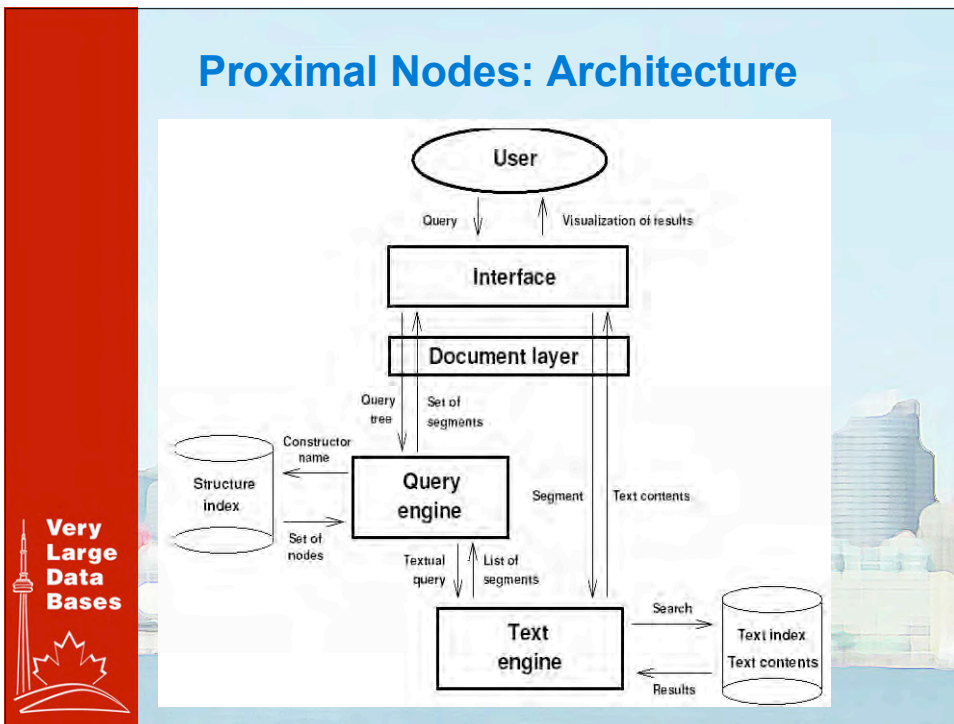
- Text = sequence of symbols (filtered)
- Structure = set of independent and disjoint hierarchies or “views”
- Node = Constructor + Segment
- Segment of node  $\supseteq$  segment of children
- Text view, to modelize pattern-matching queries
- Query result = subset of some view

## Proximal Nodes: Hierarchies









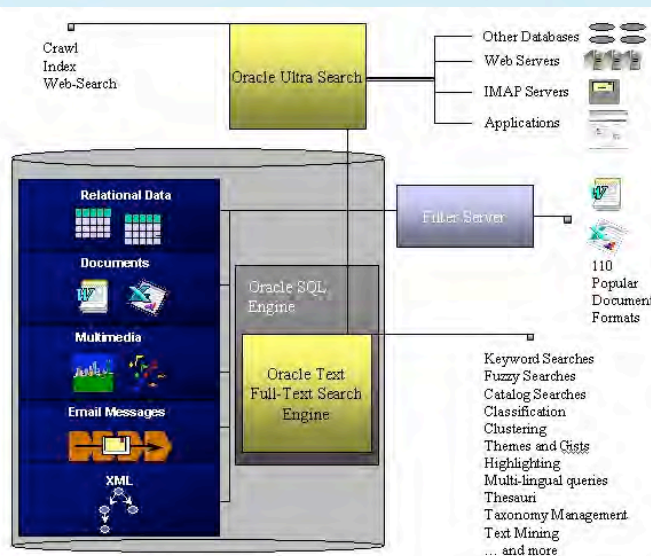
**5. Industrial DB-IR Examples: Oracle, Verity**

- DB View: Oracle
- IR View: Verity
- Provided by them!
- Thanks to
  - Omar Alonso (Oracle)
  - Prabakhar Raghavan (Verity)

## A DB Example: Oracle

- Oracle Text
  - Complete API for building any type of search application
  - Features range from basic keyword searching to advanced techniques like classification and information visualization
- Oracle Ultra Search
  - Out-of-the-box solution that requires no coding
  - Can search across OCS components, websites, databases, files, email, and Portal
  - Built on top of Oracle Text
- Included free with the standard system

## Oracle Text Search Architecture



## Common Myths about Oracle Search (according to Oracle)

- Database-Integrated Search Technology is slow
- Oracle's Search Technology is less functional than specialized search-only engines
- Major sites must run specialized search engines
- Oracle is expensive
- Oracle is complex
- Oracle's search technology will not scale out
- You can only search database-resident content with Oracle



## Oracle Text Search Functionality

- Fully integrated with the database
- Premier text search quality (TREC-8 win)
- Advanced linguistics: built-in extensible thesaurus, themes, gists, fuzzy, internationalization features for multilingual applications, etc.
- Document services: multilingual highlighting, themes, navigation ...
- XML support
- Classification (TREC-10 win)
- Statistical Text Processing: Clustering
- Integrated with JDeveloper Java IDE
- Filters for 100+ document formats
- Specialized indexes for catalogs, classification, XPath searches
- Visualization
- Integrated web-crawler and out-of-the-box-GUI with Ultra Search


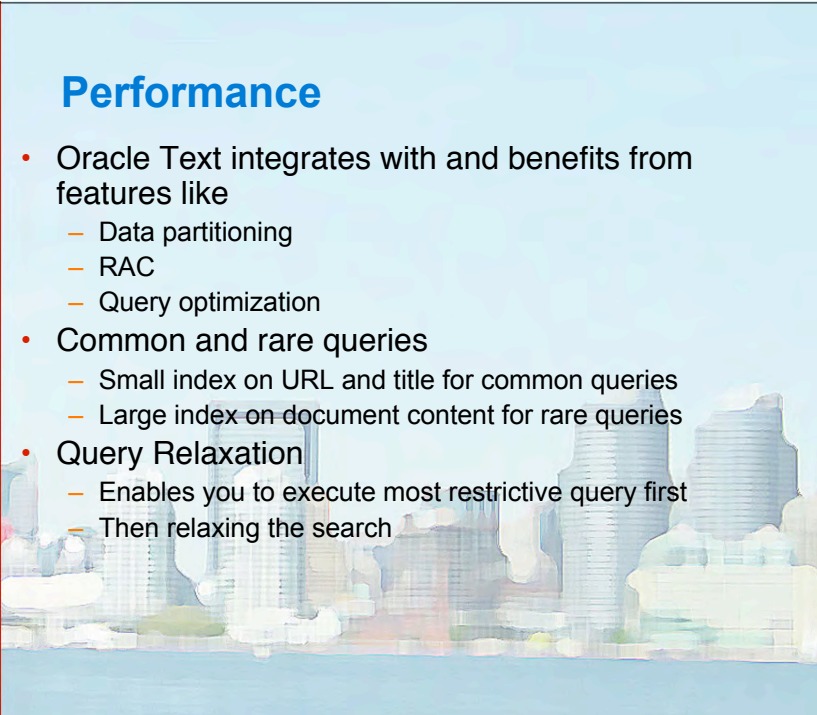






## Quality

- Link awareness
  - Popular pages and hubs
  - Website structure
  - Page structure
- Duplicate elimination
  - Remove URLs with duplicate or near duplicate content
- Spelling correction
  - Component that uses a dictionary and data from query logs
  - *Did you mean ...?*
- KWIC (Key Word In Context)
  - Highlights relevant parts of the document
  - No need to open the URL if it doesn't look relevant



## Performance

- Oracle Text integrates with and benefits from features like
  - Data partitioning
  - RAC
  - Query optimization
- Common and rare queries
  - Small index on URL and title for common queries
  - Large index on document content for rare queries
- Query Relaxation
  - Enables you to execute most restrictive query first
  - Then relaxing the search

## Ease of Use

- Users want a simple and easy to use search interface
- Hide all the complexity and expose simple interface
- Ultra Search
- Two search modes
  - Basic: simple search box where search results are sorted by relevance
  - Advanced: interface with more options where user has more control over the collection

## Personalization

- Know user search patterns
  - What do they search?
  - When do they search?
- Search query log analysis
  - Which queries were made?
  - Which queries were successful?
  - How many times was each query made?



## Advanced Features

- Classification
  - Supervised classification of content
  - Two ways: rules or training sets
  - You can group a number of categories into a taxonomy
  - Very useful for defining a common vocabulary in an enterprise
- Clustering
  - Unsupervised classification of patterns into groups
  - The engine analyzes the document collection and outputs a set of clusters with documents on it
  - Very useful for *discovering* patterns or nuggets in collections
  - Could be used as a starting point when there is no taxonomy present



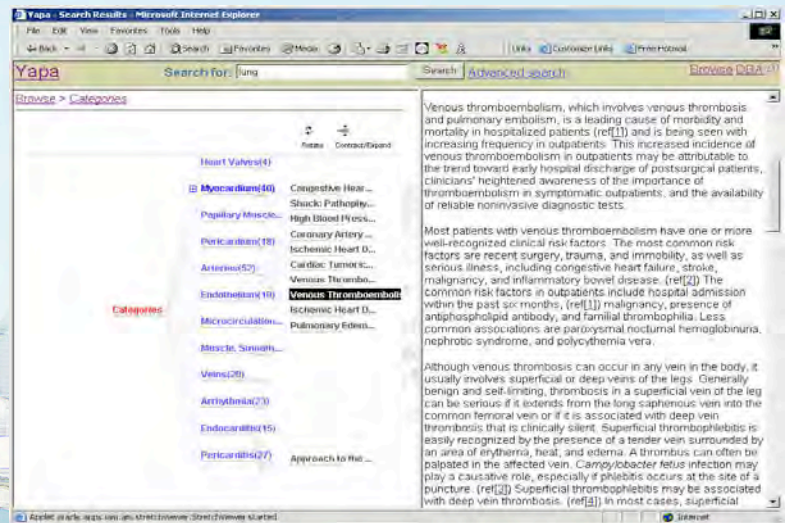
## Information Visualization

- Present searched information in ways other than hit-lists
- Shows relationship across items in addition to satisfying query results
- Better IR using visual metaphors
- Very useful for
  - Navigation through large data sets
  - Discover relationships and associations between items
  - Focus + context tasks
- Number of visualizations available
  - StretchViewer
  - Interactive Viewer (ThemeMap, Cluster visualization)
  - Integration with 3<sup>rd</sup> party vendors

## StretchViewer



Very  
Large  
Data  
Bases



## ThemeMap

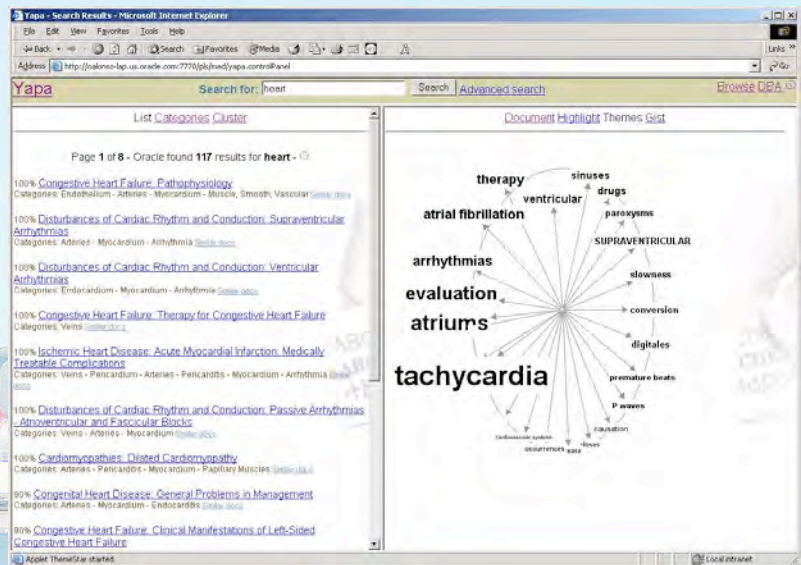


Very  
Large  
Data  
Bases





## ThemeStar



## Is Oracle's Text Search Complex?

- Easy to Develop
  - Simple SQL and PL/SQL interface
    - Can be used by any developer that knows SQL
    - Can be called by any tool that knows SQL
    - Using any language: Java, JSP, PL/SQL, C, etc.
  - Choice of datastores
    - Stored in the database
    - Stored in the file system
    - Stored on the web (URL)
    - User-defined datastore
- Easy to Deploy
- Easy to Maintain



## Oracle Text API

- Three index types
  - **context**: classic text searching
  - **ctxcat**: catalog searching
  - **ctxrule**: classification/routing applications
- Extensions to SQL
  - `select title from my_table where contains(text, 'Java') > 0;`
  - `select title from my_categories where matches(myquery, mydoc) > 0;`

## Oracle Text API – II

- Operators: Boolean expressions, phrases, proximity, fuzzy, stemming, wildcards, accumulate scores, term weighting, XPath, etc.
- Packages
  - CTX\_DOC: document services
  - CTX\_QUERY: query feedback
  - CTX\_REPORT: index information
  - CTX\_OUTPUT: logging
  - CTX\_THES: thesaurus features
  - CTX\_CLS: training set
  - CTX\_ADM: administration
  - CTX\_DDL: create/manages index preferences, sections, stop lists

## An IR Example: Verity Structured data

- Indexing databases
  - Used to import data from ODBC databases into Verity indexes (“collections”)
  - Similar to Verity gateways to other backend repositories e.g., Lotus Notes, Exchange, Documentum, Filenet, etc.
- Parametric selection for search
  - Intersect full-text search with range queries/selection
  - When a field is a taxonomy (e.g., Continent/Country/City/Street), you have relational taxonomies = Cartesian product of taxonomies

## Database indexing – 2 choices

- “Export” to XML or Bulk Insert File
- ODBC Gateway
- The common theme to either approach is to preserve the database structure in the index, such that you can query/display/sort on fields of integer, float, date, string, “attachment” data types.



## “Export” to XML or BIF - Overview

- Many applications use a database as a storage component.
- Verity may not have an official gateway to that system because the APIs may not exist and/or a simpler solution exists.
- Sample list of applications that may be indexed using this approach
  - MatrixOne, Siebel, Interwoven, Fatwire, Virage, many others
- The general concept is to temporarily export the database row/field structure in a Verity compatible format.
- A variety of integration languages have been used – including, but not limited to ASP, Java/JSP/JDBC, Perl/ODBC, etc.



## Verity Gateways

- Pre-built Gateways provide access to the most common enterprise repositories
- Gateway developer’s kit enables you to build custom gateways to virtually any application
- K2 Enterprise enforces existing security models
  - Including native security of applications accessed by Verity Gateways
  - Ensures end-users can only view the information that they are authorized to access






## Verity Gateways

### Pre-built Verity Gateways

- Available for the following repositories:
  - Documentum
  - File Systems (NFTS and UNIX)
  - HTTP
  - Lotus Notes
  - Microsoft Exchange
  - ODBC databases

### Verity Gateway Development Kit

- Quickly and easily build secure custom gateways to additional repositories



## ODBC gateway

- Verity product that uses ODBC (Data-Direct drivers) to stream records from database into Verity collections.
- A graphical tool (MMC plug-in) is used to build the text-based configurations that control the desired mapping behavior.



## ODBC GW - Certified Platforms

- Windows (with access to Oracle, DB2, Microsoft SQL Server)
- Solaris (with access to Oracle and DB2)
- AIX (with access to Oracle and DB2)
- HP-UX (with access to Oracle and DB2)
- Linux (with access to Oracle and DB2)
- Other databases such as Informix, Sybase, MySQL and others are supported
  - Gateway uses ODBC 3.5 API calls to insure compatibility



## Feature Highlights

- SQL statements that select fields from one or more tables (gateway join)
- Full Data Type support
  - Blobs, unsigned/signed integers, floats, dates
  - Filebyname – treat field as file system path and automatically follow and index
- Multi-row records
- Compound primary keys
- Efficient spidering
  - Event-driven updates – use database triggers
  - Where clauses can be used for crawling limit

## Verity K2 Enterprise Search - Parametric Selection

- Intuitive interface enables users to easily sort and filter information by selecting pre-set parameters and searching through filtered text fields and document content for specific text



## Verity K2 Enterprise Search - Parametric Selection Example

**stock Finder** Search for:  Search

Show All 3709 Matches Found 1 2 3 4 5 6 7 8 9 10 11 Next

Ticker	Sector	Industry	Daily Volume	Recent Price	Total Cash	Sales	Market Capitalization
WFLA	Financial	Regional Banks	136.0	\$12.90	\$7.63M	\$0.00K	\$16.20M
ENGFF	Capital Goods	Construction Services	162.0	\$3.53	\$10.20M	\$0.00K	\$30.40M
YIATZ	Consumer Non-Cyclical	Personal & Household Products	455.0	\$4.65	\$66.50M	\$0.00K	\$36.720M
HYWEN	Financial	S&I/Savings Banks	455.0	\$3.91	\$300.00K	\$0.00K	\$5.30M
JXSH	Financial	S&I/Savings Banks	500.0	\$10.55	\$10.00M	\$0.00K	\$20.10M
WULCF	Consumer Non-Cyclical	Food Processing	545.0	\$2.98	\$0.00K	\$0.00K	\$12.20M
ESBX	Financial	S&I/Savings Banks	591.0	\$25.20	\$5.48M	\$0.00K	\$22.90M
BAIRC	Financial	Regional Banks	664.0	\$38.75	\$26.80M	\$0.00K	\$78.70M
BMVF	Basic Materials	Iron & Steel	1,136.0	\$0.10	\$0.00K	\$0.00K	\$1.99M
IKSC	Financial	Regional Banks	1,136.0	\$12.75	\$9.54M	\$0.00K	\$32.50M
IRAGY	Financial	Mac. Financial Services	1,273.0	\$0.19	\$0.00K	\$0.00K	\$1.13M
TATF	Capital Goods	Aerospace & Defense	1,409.0	\$2.20	\$0.00K	\$0.00K	\$9.65M

**Filters:**

Daily Volume: Any  
Sector: Any  
Total Cash: Any

Market Capitalization: All Caps  
☐ Small Cap (3418) ☐ Mid Cap (209) ☐ Blue Chip (82)

Industry: Any  
 Sector: All Sectors  
☐ Basic Materials (118) ☐ Capital Goods (129)  
☐ Conglomerates (2) ☐ Consumer Cyclical (143)  
☐ Consumer Non-Cyclical (79) ☐ Energy (72)  
☐ Financial (710) ☐ Healthcare (341)  
☐ Services (509) ☐ Technology (1194)  
☐ Transportation (58) ☐ Utilities (16)

Select

## Verity K2 Enterprise Search - Relational Taxonomies

- Allows users to quickly narrow down information in the way that makes the most sense to them
  - Users take alternate paths through the same topics or categories to quickly and easily narrow down on the information they need
  - Users can navigate to information using two or more taxonomies at once
- Dramatically improve the finding experience for data with attributes



**carFinder**  
Thousands of quality used vehicles!

Search for:

**Browse**

**Geography:** Australia (2177) Canada (1518) U.S.A. (2413) Asia (4297) European (2159) North American (4348)  
 Brisbane (182) Gold Coast (411) Calgary (192) Montreal (382) California (2329) Florida (1621) Acura (1124) Scion (8) Alfa Romeo (77) Aston Martin (71) All General (77)

**Filter by:**

**Category:** Any  
**Price:** Any  
**Mileage:** Any  
**Year:** Any

**Color:** All Colors  
☐ Black (2603) ☐ Blue (996) ☐ Gold (996)  
☐ Green (976) ☐ Red (1029) ☐ Silver (1115)  
☐ White (1894) ☐ Yellow (779)

**Select**

**12900 Matches Found** 1 2 3 4 5 6 7 8 9 10 11 Next

Sort results by:	Category	Color	Year	Price	Mileage	Details
1	Compact	Yellow	1998	7500	14900	<a href="#">Details</a>
2	Compact	White	1993	7200	15000	<a href="#">Details</a>
3	Compact	White	2001	7300	14100	<a href="#">Details</a>
4	Compact	White	1999	7400	15000	<a href="#">Details</a>
5	Compact	Green	2001	7500	14100	<a href="#">Details</a>
6	Compact	Gold	1996	7100	14300	<a href="#">Details</a>
7	Compact	Blue	1995	7200	14400	<a href="#">Details</a>

©2002 Verity, Inc.  
 CarFinder is a demonstration tool for Verity parameter search. Any similarity to actual companies or products is purely coincidental. The products advertised on carfinder.com are not available for sale. To purchase Verity parameter search, please contact Verity, Inc.



## 6. DB Approaches

- IR on Relational Data
- IR on XML
  - Content Only (CO)
  - Content and Structure (CAS)



## IR on Relational Data

- DBXplorer [Agrawal et al.]
- DISCOVER [Hristidis et al.]
- BANKS [Hulgeri et al.]




## IR on XML: XSearch

```
<proceedings>
  <inproceedings>
    <author>Moshe Y. Vardi</author>
    <title>Querying Logical Databases</title>
  </inproceedings>
  <inproceedings>
    <author>Victor Vianu</author>
    <title>A Web Odyssey: From Codd to
XML</title>
  </inproceedings>
</proceedings>
```

## The CO Approach

Find papers by Vianu on the topic of  
“logical databases”

**Search:** Vianu logical databases 

- Each **document** in the corpus is treated as a **unit**.
- A document containing some of the three query terms is considered as a result

The document contains the three query terms.  
Hence, it is returned by a standard search engine. **BUT**

~~This does not work!!!~~

```
<proceedings>
  <inproceedings>
    <author>Moshe Y. Vardi</author>
    <title>Querying Logical Databases</title>
  </inproceedings>
  <inproceedings>
    <author>Victor Vianu</author>
    <title>A Web Odyssey: From Codd to
      XML</title>
  </inproceedings>
</proceedings>
```

## XQuery+FT Query Language

```
FOR $i IN document("bib.xml")//inproceedings
WHERE $i/author contains 'Vianu'
  AND $i/title contains 'Logical'
  AND $i/title contains 'Databases'
RETURN <result>
  <author> $i/author </author>
  <title> $i/title </title>
</result>
```

**This does work, BUT**

- Much more complicated query expression than search box
- Extensive knowledge of the document structure is required to write the query
- Still need to choose a mechanism for ranking the results



## Requirements from the Search Tool

- A simple syntax that can be used by **naive users**
- Search results should include XML **fragments** and not necessarily full documents
- The XML fragments in an answer, should be **semantically related**
  - For example, a paper and an author should be in an answer only if the paper was written by this author
- Search results should be **ranked**
- Search results should be returned in **“reasonable” time**

## XSearch Query Syntax

- A **query** is a list of *query terms*
- A query term can be a
  - **Keyword**, e.g., **database**
  - **Tag**, e.g., **inproceedings:**
  - **Tag-keyword** combination, e.g., **author:Vianu**
- Optionally preceded by a ‘+’



## The Example Revisited

- Find papers by Vianu on the topic of “logical databases”

logical +database inproceedings: author:Vianu

The keyword database of Vianu under the  
must appear in the fragment in the fragment,  
t increases increases the rank of this fragment

**XSearch:** author:Vianu title:

```
<proceedings>
  <inproceedings>
    <author>Moshe Y. Vardi</author>
    <title>Querying Logical Databases</title>
  </inproceedings>
  <inproceedings>
    <author>Victor Vianu</author>
    <author>Victor Vianu</author> </title>
  </inproceedings>
  <title>A Web Odyssey: From Codd to XML</title>
</proceedings>
```

**Good Result!**

title and author elements ARE semantically related

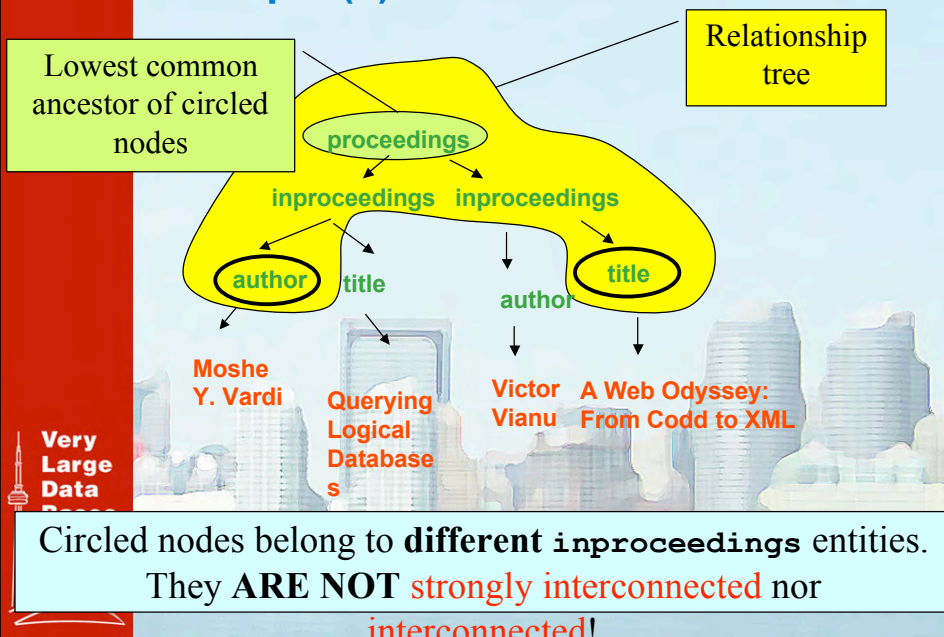
**XSearch:** author:Vianu title:

```
<proceedings>
  <inproceedings>
    <author>Moshe Y. Vardi</author>
    <title>Querying Logical Databases</title>
  </inproceedings>
  <inproceedings>
    <author>Victor Vianu</author>
    <title>A Web Odyssey: From Codd to XML</title>
  </inproceedings>
</proceedings>
```

### Bad Result!

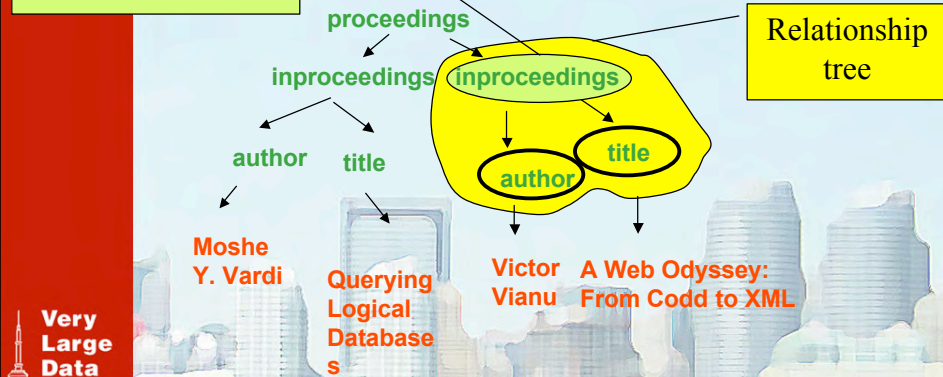
title and author elements ARE NOT semantically related

### Example (1)



## Example (2)

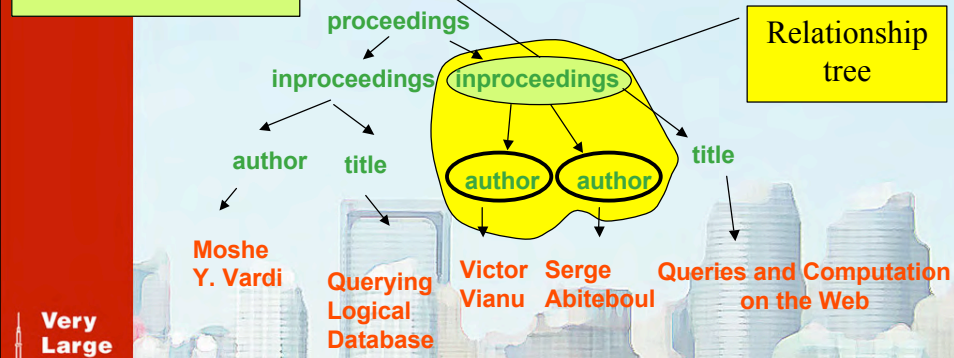
Lowest common ancestor of circled nodes



Circled nodes belong to the **same inproceedings** entity.  
They **ARE strongly interconnected**, thus, **interconnected**!

## Example (3)

Lowest common ancestor of circled nodes



We can see the advantage of using **interconnection** rather than **strong interconnection**.  
These two **author** nodes ARE semantically related.

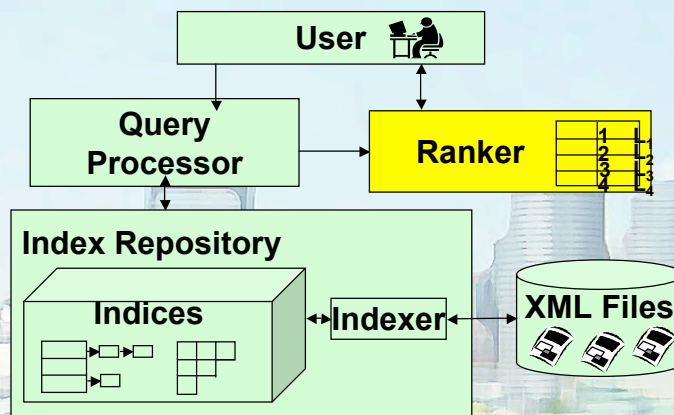
## Query Processing

Document fragments are **extracted** using the interconnection index and other indices

Extracted fragments are **returned ranked** by the estimated relevance



## Ranker





## Ranking Factors

Several factors increase the rank of a result

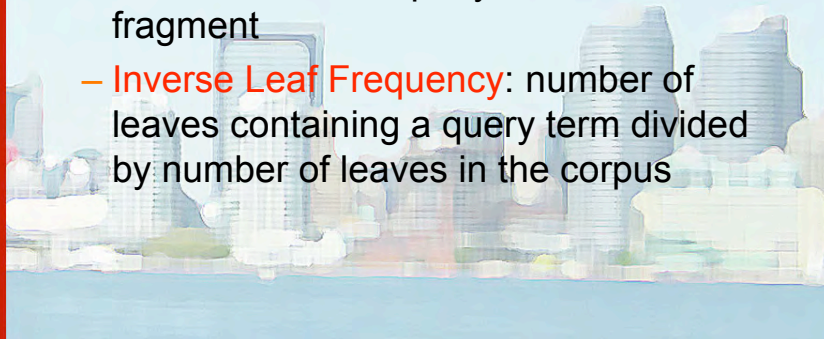
- **Similarity** between query and result
- **Weight of labels** appearing in the result
- **Characteristics** of result tree



## Query and Result Similarity

### TF-ILF

- Extension of TFIDF, classical in IR
- **Term Frequency**: number of occurrences of a query term in a fragment
- **Inverse Leaf Frequency**: number of leaves containing a query term divided by number of leaves in the corpus



## TF-ILF

- Term frequency of keyword  $k$  in a leaf node  $n_l$

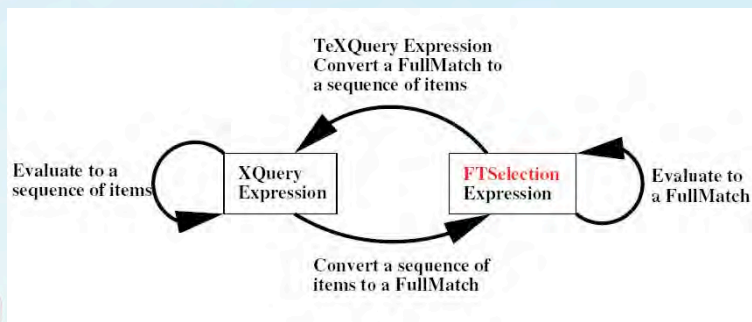
$$tf(k, n_l) := \frac{occ(k, n_l)}{\max\{occ(k', n_l) \mid k' \in words(n_l)\}}$$

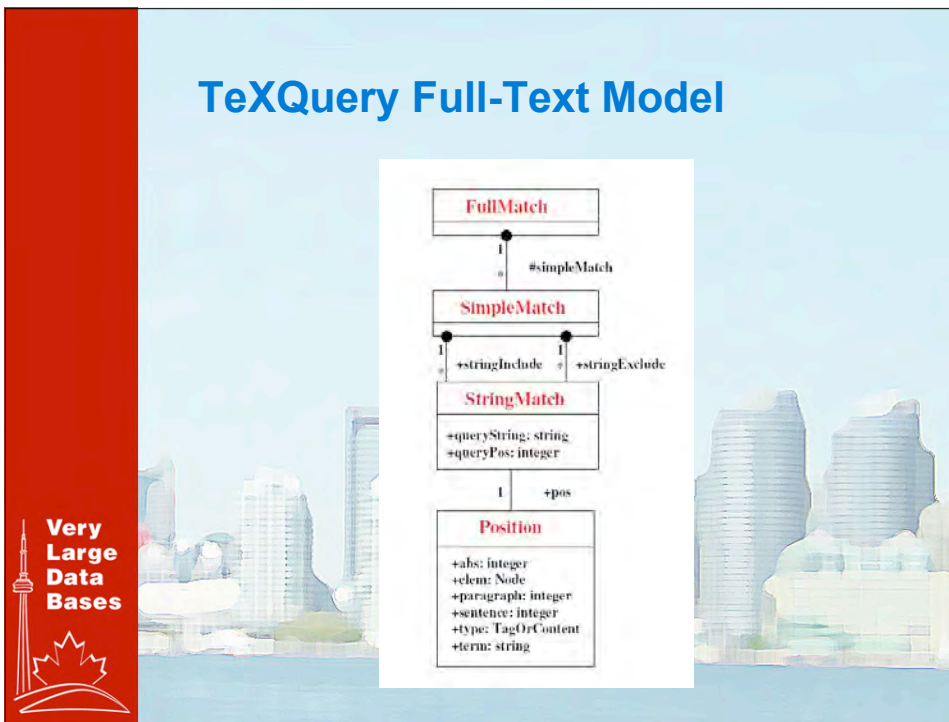
- Inverse leaf frequency


$$ilf(k) := \log \left( 1 + \frac{|N|}{|\{n' \in N \mid k \in words(n')\}|} \right)$$

**TFILF** is the product between *tf* and *ilf*

## IR on XML: TeXQuery

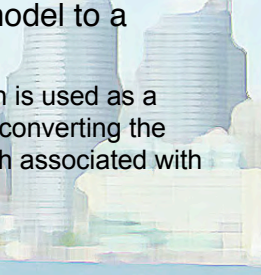




**Very  
Large  
Data  
Bases**


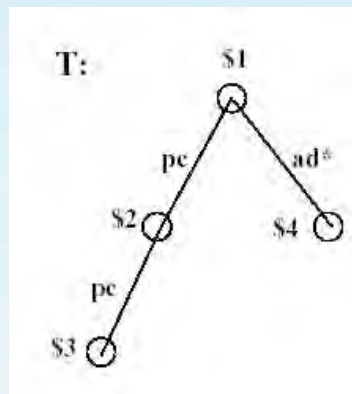
## TeXQuery Composability

- Conversion from FullMatch to XQuery data model:
  - To enable TeXQuery expressions to be nested and composed with regular XQuery expressions three new XQuery expressions are proposed for FTS which convert a FullMatch to a sequence of items
- Conversion from XQuery data model to a FullMatch:
  - The result of an XQuery expression is used as a search token in an FTSelection by converting the XQuery expression to the FullMatch associated with that search token



## IR on XML: TIX Algebra

- TIX (Text In XML) algebra, is based on the idea of a “*scored tree*”. Each operation in this algebra manipulates collections of scored trees.
- By using this algebra, deciding the granularity of the elements to be returned to the user is possible.
- The “*Pick*” operator in TIX, it is possible to select appropriate elements based on some relevance criteria among the others in an XML document



Find document components in articles.xml that are part of an article written by an author with last name “Doe” and are about “search engine”. Relevance to “internet” and “information retrieval” is desirable but not necessary.

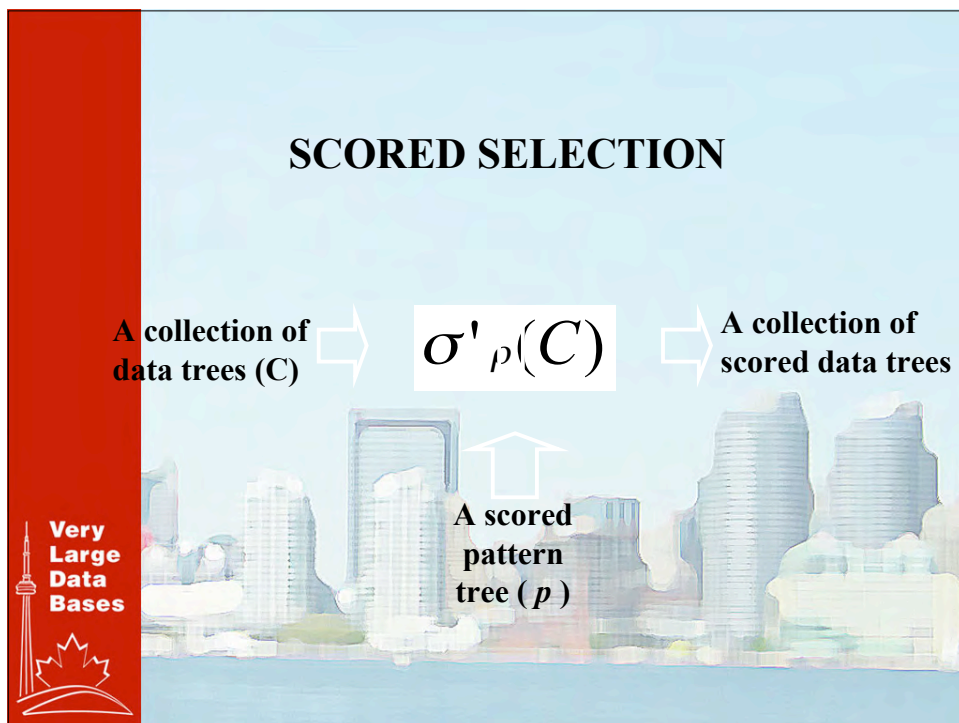
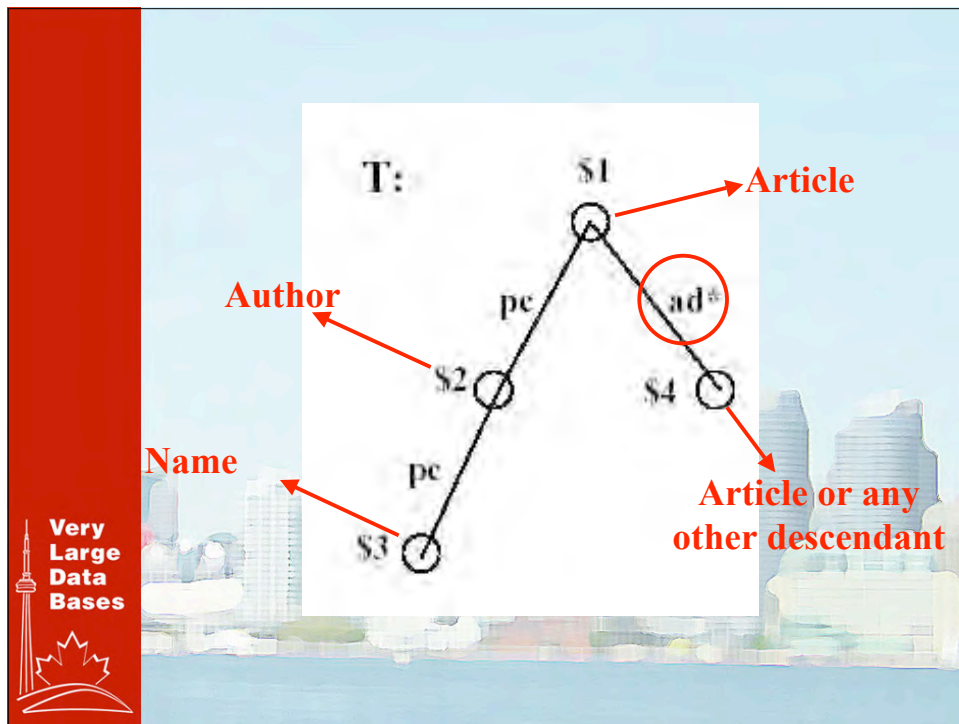
F: (\$1.tag = article) & (\$2.tag = author) & (\$3.tag = sname) & (\$3.content = "Doe")

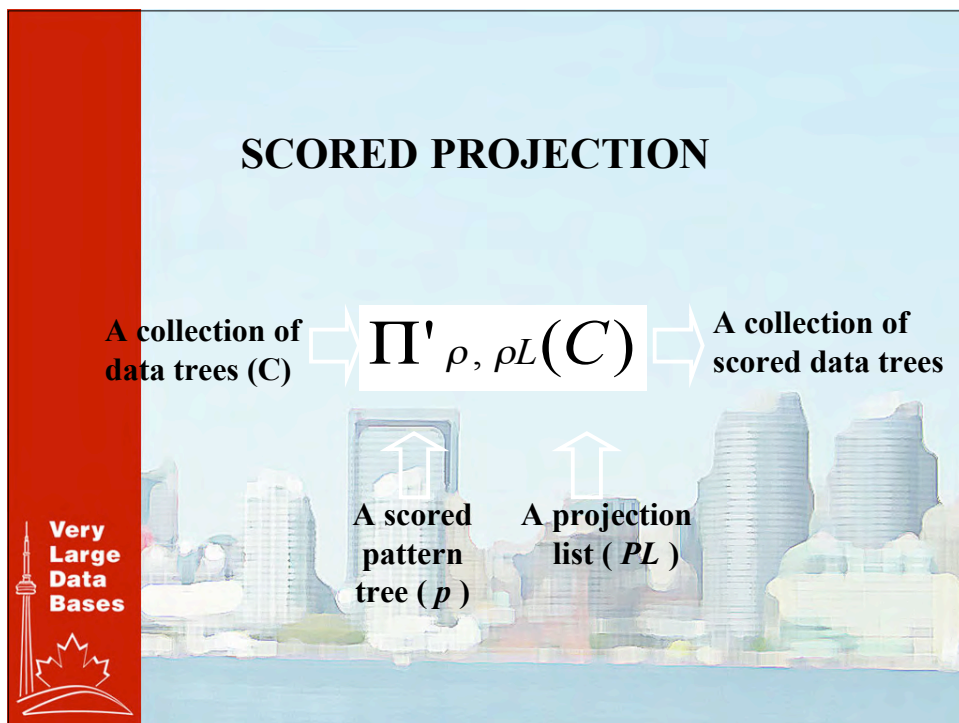
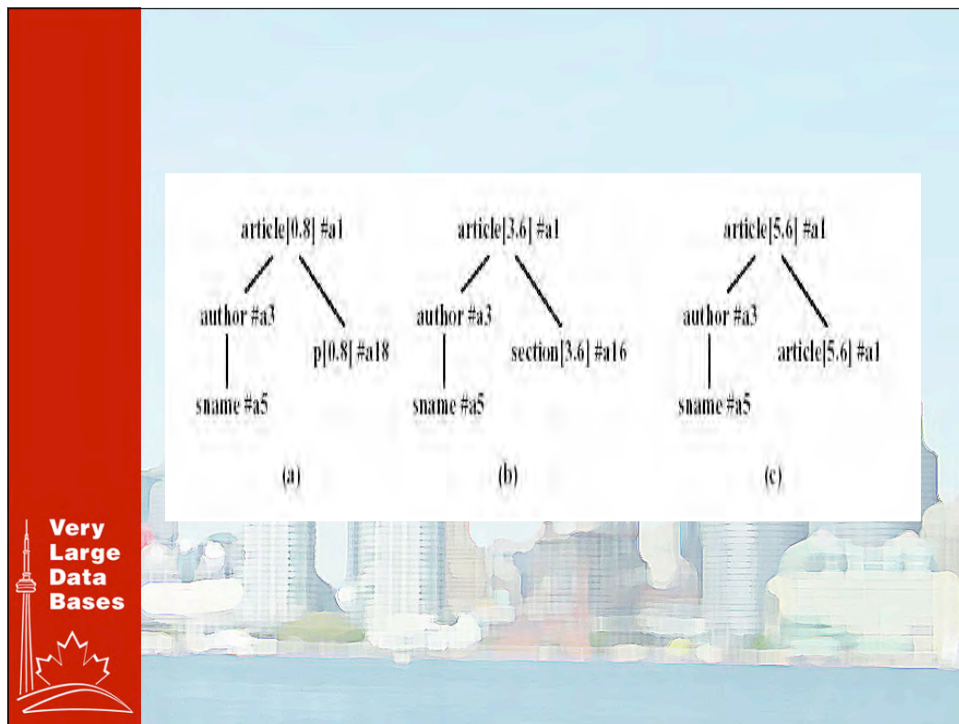
S:

\$4.score = {ScoreFoo({"search engine"}, {"internet", "information retrieval"})}

\$1.score = \$4.score







## IR-style join query

Find relevant document components in articles.xml as specified in Query below

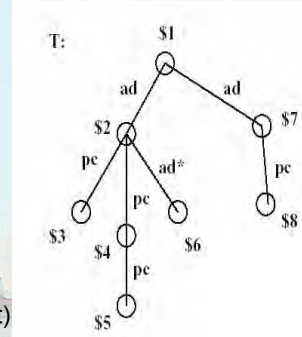
For articles containing such components, find reviews from reviews.xml for articles with similar titles.

**F:**

(\$2.tag = article) & (\$3.tag = article-title) &  
(\$4.tag = author) & (\$5.tag = sname) &  
(\$5.content = "Doe") & (\$7.tag = review) &  
(\$8.title = title) & (\$1.tag = titx\_prod\_root)

**S:**

\$6.score = ScoreFoo({"search engine"},  
{"internet", "information retrieval"})  
\$2.score = \$6.score  
\$joinScore = ScoreSim(\$3.content, \$8.content)  
\$1.score = ScoreBar(\$joinScore, \$6.score)



## 7. Hybrid & IR Approaches

- Overview of Approaches
- Retrieval Models
- Indexing
- INEX
- Ranking XML


## Overview of Approaches

- RBD + IR: Two different APIs
- RDB + IR Hybrid: QUIQ, MOA, HySpirit, ...
- RBD “text search” accelerator
  - Text content is transformed to flat XML
  - XML is searched using an IR API
  - Results can be later combined with SQL
- IR System with SQL support
  - Special indexes for atomic data types
- XML Databases
  - Atomic data types as attributes (metadata)
  - Implementation on top of structured text models?

## QUIQ (Kabra *et al*, 2003)

- Tuple: <tag-name, tag-type, tag-value>
- Query: *match-filter-quality*
  - Result: AND of *match* & *filter*
  - *Match* are approximate constraints
  - *Filter* are exact constraints
  - Relevance is adjusted by *quality*
- Indexing: built on top of a RDBMS
  - Non-text data is mapped to pseudo-words
  - Unified index & common TF-IDF model
  - Deferred update operations
- Evaluation: 60% faster than a RDBMS text extension






Very  
Large  
Data  
Bases

## Retrieval Models

- Relational Model: DB2XML, XML-QL, TSIMMIS, LOREL
- Object-oriented Model: SOX, StruQL, ...
- Extended Vector Model
- Weighted Boolean Model: XQL, ...
- Probabilistic Model: XIRQL, ELIXIR, JuruXML, ...



Very  
Large  
Data  
Bases

## Indexing

- Flat File: add information, SQL accelerators,...
- Semi-structured:
  - Field based: no overlapping, Hybrid model,...
  - Segment based: Overlapped list, List of references, p-strings
  - Tree based: Proximal Nodes, XRS, ...
- Structured:
  - IR/DB, Path-based, Position-based, Multidimensional
- Indexes:
  - Structure + Value index:
    - Toxin, Dataguides, T-indexes, Index Fabric, etc.
  - Integrated Full-text and Structure index:
    - Proximal Nodes, Region Algebra, String Indexing, ...

## XPath over Proximal Nodes (Navarro & Ortega, 2003)

- A fast implementation of XPath subset
- Maps XPath expressions into Proximal Nodes algebra
- Format translation of Axes
- Node + Text index
- Lazy evaluation

Query	IXPN	Xind	eXist	Grep	Saxon	MS	Toxin
/tstmt/bookcoll/book/c	1.8	20.5	8.8	3.4	4.0	3.3	2.5
hapter	0.5	2.8	2.2	0.7	3.3	1.3	-
/tstmt/coverpg/coverpg	1.8	58.9	8.8	3.8	4.1	3.2	2.5
[title]	0.9	22.7	8.8	3.7	4.0	4.2	-
/tstmt[//chapter	0.4	9.9	9.8	0.7	3.4	1.8	3.7
v[.="love" ]							
/tstmt[coverpg/title							
/following-							
silbling: :subtitle	0.5	2.6	9.8	0.7	3.3	1.3	-

## INEX

- Initiative for the Evaluation of XML
- Three types of tasks:
  - Content only search
  - Content & Structure Search
  - Clustering
- Started in 2002
- Cooperative relevance assesment
- About 40 groups per year

## Ranking XML

- Content only:
  - exploit hierarchical structure
  - exploit importance of tags
- Content & structure:
  - Query languages with uncertainty & vagueness
  - Data types with vague predicates
  - Strict & fuzzy structural conditions
  - Dynamic  $tf \times idf$

## Integrated IR (Bremer & Gertz)

- Extension to XQuery
- Based on XML fragments
- Schemas are extended DataGuides
  - Enumeration of all rooted label paths
- Ancestor relationships from structural joins
- RANKBY operator
  - based on local & dynamic  $tf-idf$
- New node enumeration encoding
- Path & term-index
  - Other smaller indexes (in total less than 60%)
- More than 10 times faster than other XQuery prototypes

## 8. Open Problems

- Heterogenous data
- Ranking tuples & XML
- Indexing & searching
- New retrieval models
- DB issues for documents
- Efficient algorithms
- Optimization and algebras
- Quality evaluation

## 9. Bibliography – 1

- Baeza-Yates & Ribeiro-Neto, Modern Information Retrieval, Addison-Wesley, 1999.
- Baeza-Yates & Navarro, Integrating contents and structure in text retrieval, SIGMOD 25 (1996), 67-79.
- Baeza-Yates and Navarro, XQL and Proximal Nodes, JASIST 53, 504--514, 2002.
- Baeza-Yates, Carmel, Maarek, and Sofer, editors. Special issue on XML Retrieval, JASIST, 53, 2002.
- Baeza-Yates, Fuhr, and Maarek, editors. Proceedings of the SIGIR 2002 Workshop on XML and Information Retrieval.
- Bremer & Gertz, Integrating Document & Data Retrieval Based on XML, to appear.
- Chinenyanga and Kushmerik, Expressive retrieval from XML documents, Proc. of the 24th SIGIR, 163-171, New York, 2001.
- Delgado & Baeza-Yates, A Comparison of XML Query Languages, Upgrade 3, 12-25, 2002.



## Bibliography - 2

- Fuhr and Grossjohann , XIRQL: An XML query language based on IR concepts. ACM TOIS 22, 313--356, 2004.
- Fuhr, Govert, Kazai, and Lalmas, editors. INitiative for the Evaluation of XML Retrieval. Proceedings of the First INEX Workshop. Dagstuhl, Germany, Dec., 8--11, 2002
- Fuhr, Lalmas, and Malik, editors. Proc. of the Second INEX Workshop. Dagstuhl, Germany, Dec. 15--17, 2003, 2004.
- Grabs and Schek, Flexible information retrieval from XML with PowerDB-XML, In INEX 2003, 141-148.
- Kabra, Ramakrishnan, Ercegovac, The QUIQ Engine: A Hybrid IR DB System, ICDE 2003.
- 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. In INEX 2003, 73-90.

## Bibliography - 3

- Mihajlovic, Hiemstra, Block & Apers, An XML-IR-DB Sandwich: Is it better with an Algebra in between?, I Workshop on DB-IR integration at SIGIR, 2004.
- Navarro and Ortega, IXPn: An index-based XPath implementation, Technical Report, U. de Chile, 2003.
- Piwowarski, Vu, and Gallinari. Bayesian networks and INEX 2003. In INEX 2004.
- Sayyadian, Shakery, Doan & Zhai, Toward Entity Retrieval over Structured and Text Data, I Workshop on DB-IR integration at SIGIR, 2004..



## Bibliography – COMPLETE

- [ACS02] S. Amer-Yahia, S. Cho, and D. Srivastava. Tree pattern relaxation. In *Proceedings of the 8th Conference on Extending Database Technology (EDBT)*, pages 496-513, Prague, Czech Republic, March 2002.
- [ACD02] S. Agrawal, S. Chaudhuri, G. Das, DBXplorer: A System for Keyword-Based Search over Relational Databases, ICDE Conf., 2002.
- [AQM+97] S. Abiteboul, D. Quass, J. McHugh, J. Widom, and J. Wiener. The Lorel query language for semistructured data. *International Journal on Digital Libraries*, 1(1):68-88, 1997.
- [An03] A. Aron et al. XQueC: Pushing queries to compressed XML data. In *VLDB*, 2003.
- [AYJ03] S. Al-Khalifa, C. Yu, and H. V. Jagadish. Querying structured text in an XML database. In *SIGMOD*, 2003.
- [BCF+02] S. Boag, D. Chamberlin, M. F. Fernandez, D. Florescu, J. Robie, J. Simeon, and M. Stefanescu, editors. XQuery 1.0: An XML query language. W3C Working Draft, April 2002. <http://www.w3.org/TR/xquery>.
- [BGK03] P. Buneman, M. Grohe, and C. Koch. Path queries on compressed XML. In *VLDB*, 2003.
- [BKS02] N. Bruno, N. Koudas, and D. Srivastava. Holistic twig joins: Optimal XML pattern matching. In *SIGMOD*, 2002.
- [BR99] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley, Harlow, UK, 1999.
- [Bur92a] F. Burkowski. An algebra for hierarchically organized text-dominated databases. *Information Processing and Management*, 28(3):333(348), 1992.
- [Bur92b] F. Burkowski. Retrieval activities in a database consisting of heterogeneous collections of structured texts. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 112(125, Copenhagen, Denmark, June 1992.
- [CAC94] V. Christophides, S. Abiteboul, S. Cluet, and M. Scholl. From structured documents to novel query facilities. In *Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data*, pages 313-324, Minneapolis, Minnesota, USA, June 1994.
- [CCB95a] C. Clarke, G. Cormack, and F. Burkowski. An algebra for structured text search and a framework for its implementation. *The Computer Journal*, 38(1):43(56), 1995.
- [CCB95b] C. Clarke, G. Cormack, and F. Burkowski. Schema-independent retrieval from heterogeneous structured text. In *Proceedings of the 4th Annual Symposium on Document Analysis and Information Retrieval*, pages 276(283, London, UK, 1995.