University of
Twente

Information Retrieval Modeling

Russian Summer School in Information Retrieval

Djoerd Hiemstra
http://www.cs.utwente.nl/~hiemstra

University of Twente
The Netherlands

PART 4
Structured Information Retrieval

Overview

1. implicit vs. explicit structure
2. static vs. dynamic structure
3. multiple hierarchies
4. PF/Tijah

Course material

- Djoerd Hiemstra and Ricardo Baeza-Yates, “Structured Text Retrieval Models'', In M. Tamer Özsu and Ling Liu (eds.) *Encyclopedia of Database Systems, Springer*, 2009

Structured IR tasks

1. Content-only:
   - Search data without knowing its structure.
   - The system needs to identify the most appropriate element type for retrieval.
2. Content-and-Structure
   - Search data knowing its structure.
   - “give me articles of which the author is named '*Pavel*', and the acknowledgements contain '*University of Twente*”

Explicit structure

- Database is “well-formed” (e.g. XML)
- Simply ask for pre-defined elements

```
<SECTION> CONTAINING "HELLO"
```

(Burkowski 1992)

Implicit structure

- Free-form structure
  (e.g. old HTML versions)
  – Elements are constructed at query time

```
<SECTION> FOLLOWEDBY </SECTION>
  CONTAINING "HELLO"
```

  – No difference between word tokens and markup tokens
  – Might consider nesting, or not...

(Clarke et al. 1995; Jaakkola & Kilpelainen 1999)

Implicit structure

- Nesting or not nesting?
  - `<SECTION> FOLLOWEDBY </SECTION> CONTAINING "HELLO"`
  - `"TO" FOLLOWEDBY "BE" CONTAINING "NOT"`

Dynamic structure

- Query might add new structure
  - *p*-strings model (Gonnet & Tompa, 1987)
  - Element construction in XQuery

p-strings

- This is a database(!)

  John Doe, "Crime", Police 6, 2028.

- This is its schema:

-
```
E := { entry   :=   author ', ' title ', ' journal ', ' year '.'
       author  :=   text ;
       title   :=   ' " ' text  ' " ' ;
       journal :=   text '  ' digit+ ;
       year    :=   digit digit digit digit ;
       text    :=   ( letter | ’ ’ ) + ;  }
```

p-strings

- New grammar rule ...

```
 NameG := {
    name := ( givenname ' ' )+ surname ;
    givenname := letter + ;
    surname := letter + ;
}
```

- … used as:

(author in E) reparsed by NameG

XQuery

- XQuery
  - “FLWOR expressions”

```
For         $page in doc(“x.xml”)/html
Let         $nr_of_p := count($page//p)
Where       $nr_of_p > 10
Order by    $nr_of_p  descending
Return      <mytitle>
             { $page/head/title }
            </mytitle>
```

XPath

XPath

- //html
  - (give me all XML elements called 'html')
- //html/head/title
  - (give me all XML elements called 'title', with a 'head' parent that have a 'html' parent)
- //html[./head/title]
  - (give me all XML elements called “html” that have a “head” element with a title element)

Multiple hierarchies

- Each hierarchy serves different purpose
  - Logical structure (chapters, sections,...)
  - Lay-out structure (column 2, page 5,...)
  - Linguistic structure (noun phrase, verb,...)
- Across hierarchies elements may partially overlap

```
$doc//paragraph[./select-narrow::Verb ftcontains
"killed" and./select-narrow::person ftcontains
"Abraham Lincoln" ]
```

(Alink 2005)

Challenge:

- How to rank results of structured queries?
  - First retrieve using structure, then rank using keywords only?
  - Relevance propagation / aggregation
  - Algebraic approaches

Today: Structured IR = XML IR

- XPath
  - Explicit / single hierarchy / static
  - NEXI: simple IR extension
  - XPath Full-Text:
- XQuery
  - Explicit / single hierarchy / dynamic
  - XQuery Full-Text

Challenge

- How to combine this with ranking?
  - Done in PF/Tijah

PF/Tijah

Aims of PF/Tijah

- The system aims to be a light-weight general tool box for information retrieval

  - out of the box solutions for common tasks
  - It allows the search system developer to hook in at several levels: e.g. region algebra / or MIL (database scripting)

PF/ Tijah's Inverted file index for XML

<html>^{1}
<title>^{2} Hello^{3} world^{4} </title>^{5}
<p>^{6} some^{7} hello^{8} </p>^{9}
<p>^{10} some^{11} world^{12} </p>^{13}
</html>^{14}

<html>	(1, 14)
<title>	(2, 5)
<p>	(6, 9), (10, 13)
hello	3
world	4, 12
some	7, 11
:	:

NEXI

- Narrows Extended XPath I
  - narrowed: only descendent steps (and self)
  - extended: special **about()** function providing ranked results

```
//Article[about(.//title,search)]//Abstract[about(.,XML)]
```

in Burkowski's "algebra for contiguous extents":

```
(<Abstract> containing "XML") containedby (<Article>
  containing (<title> containing "search") )
```

What a weird name...

PATHFINDER

- **Language:** XQuery. Precise structural querying and XML generation
- **Output:** XML
  -
- **Data Model:** pre/size encoding of nodes. Text-nodes are maintained as single strings
- **Architecture:** Layered query processing generating MIL. Execution on MonetDB

TIJAH

- **Language:** NEXI. Content and structure ranking
- **Output:** Ranked sequences of scored nodes
- **Data Model:** region model with start-end encoding of words and nodes
- **Architecture:** Layered query processing generating MIL. Execution on MonetDB

Joins on values

- Find figures that describe the Corba architecture and the paragraphs that refer to those figures:

```
let $doc := doc("inex.xml")
for $p in tijah:query($doc, "//p[about(., corba)]")
for $fig in $p/ancestor::article//fig
where $fig/@id = $p//ref/@rid
return <result> { $fig, $p } </result>
```

Features of PF/Tijah

What makes PF/Tijah different from other search engines?

1. It supports retrieving arbitrary parts of textual data. No notion of “documents” at indexing time
2. It supports complex scoring of structure and content with NEXI queries
3. Enables ad hoc result presentation by means of its query language
4. Combines Text Search with possibilities of XQuery database querying

Functional embedding of NEXI in XQuery

How to call text-ranking within XQuery?

- The text ranking extension has to fit in functional XQuery language: being fully compositional with other XQuery expressions
  1. Extending the XQuery language (e.g. as proposed by the W3C's XQuery Full-Text standard)
  2. Using NEXI directly inside regular XQuery functions, since they proved to be useful for content and structure queries

How to return nodes and scores?

- Problem: Simple first-order functions cannot return nodes and scores at the same time

Functional embedding of NEXI in XQuery (2)

A set of 3 functions:

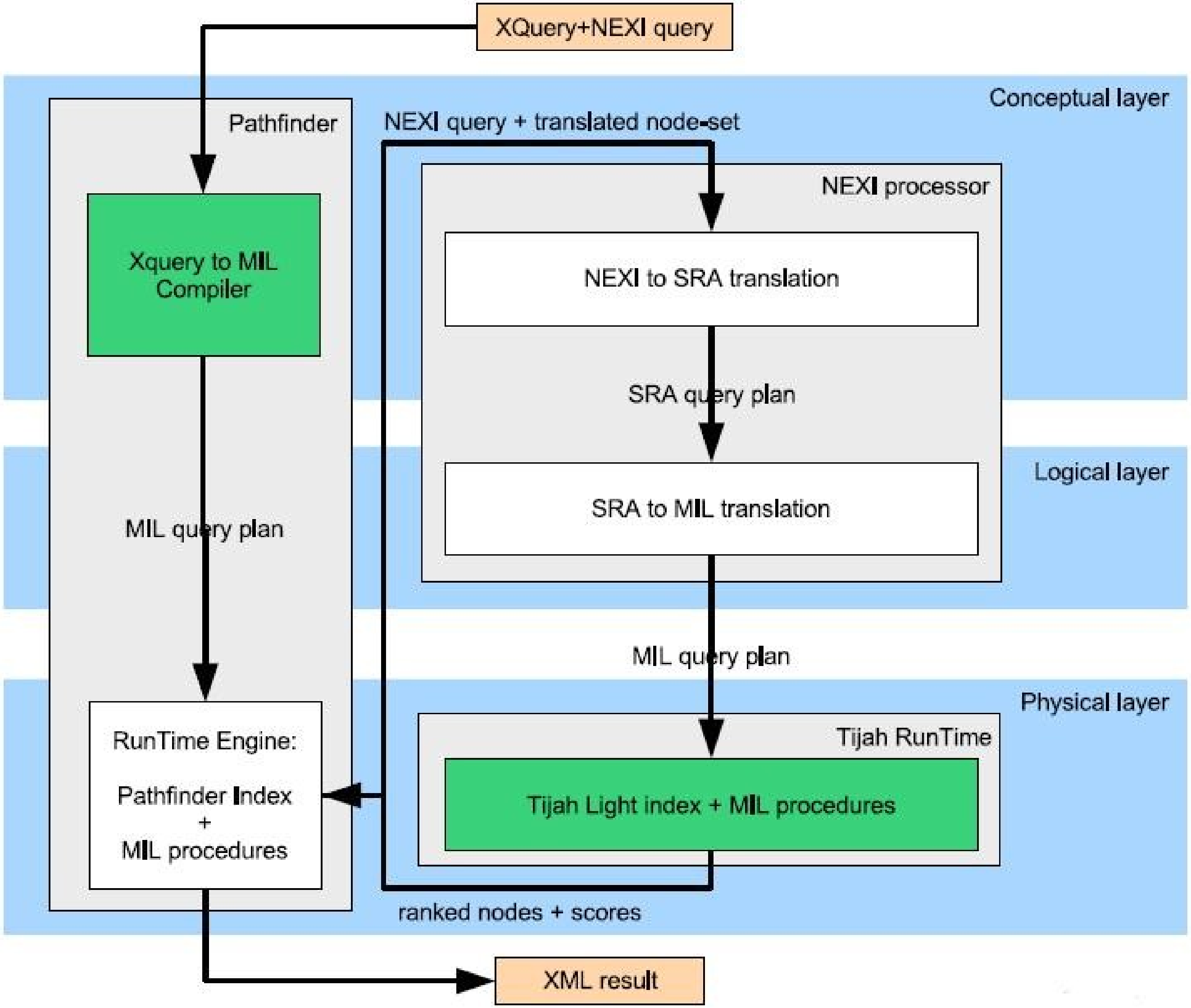
- `tijah:query-id(node-seq, "NEXI query")`returning a query identifier only
- `tijah:nodes(query-id)` returns a ranked list of nodes
- `tijah:score(query-id, node)` returns the score of that node

And one shortcut:

- `tijah:query(node-seq, "NEXI query")`

  equals
- `tijah:nodes(tijah:query-id(node-seq, "NEXI query" ))`

Integration work



Example

- Search for paragraphs about XQuery in html documents about information retrieval and databases:

```
let $c := doc("mydata.xml")
  return
  tijah:query($c,"//html[about(., ir db)]//p[about(., xquery]")
```

- XQuery FT Version:

```
let $c := doc("mydata.xml")
   for $res score $s in
   $c//html[. ftcontains ("ir", "db")]//p[. ftcontains
   "xquery"]
   order by $s descending
   return $res
```

Options

- To parameterize the search we allow options to be set in a single empty TijahOptions node:

```
let $opt := <TijahOptions ir-model="NLLR"/>
let $c := doc("mydata.xml")
for $res in tijah:query($opt, $c, "//html[about(., xml)]")
return $res//title
```

- This option node can also be loaded from a file.

Joins on values

- Find figures that describe the Corba architecture and the paragraphs that refer to those figures:

```
let $doc := doc("inex.xml")
for $p in tijah:query($doc, "//p[about(., corba)]")
for $fig in $p/ancestor::article//fig
where $fig/@id = $p//ref/@rid
return <result> { $fig, $p } </result>
```

The full-text index

What information do we need:

- Pre-order position of words and nodes
- Size of nodes for structural query constraints

For faster node selection:

- Encode terms/tags by their TID
- Building inverted posting lists for Tags and Terms

Index Overview

PF-light Index

$< \mathbf{PRE} | \mathit{size} | \mathit{TID} >$

Global Tag/Term Dictionaries

$< \mathbf{tag\text{-}/term} | \mathit{TID} >$

Inverted Indices for Tags/Terms

$< \mathbf{TID} | \mathit{PREs} >$

Overview of the Scoring Procedure

Input:

- sequence of nodes to be scored
- sequence of term occurrences in the collection

Output:

- sequence of ranked nodes and corresponding scores

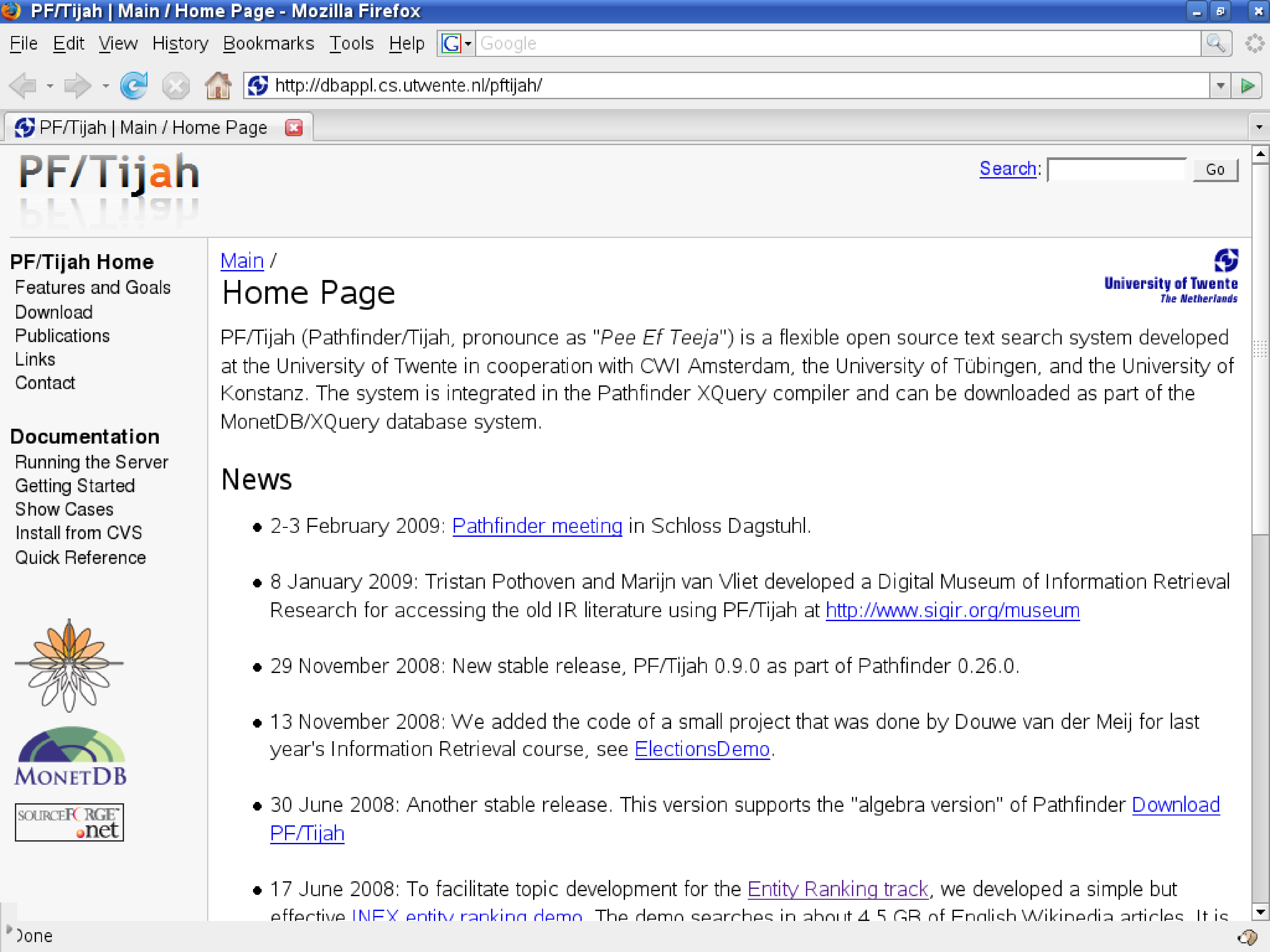
Processing Steps:

1. Get node-term pairs with containment join .
2. Aggregate and compute scores depending on the retrieval model

Current short-comings

Problems

- Database back-end needs to hold index in main memory
- Implementation of more out-of-the-box tools necessary, e.g. phrase search
- Overlapping Expressiveness of NEXI and XQuery
- String Embedding of NEXI queries remains black box to Pathfinder. No static type checking, full query compilation possible.


PF/Tijah | Main / Home Page - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Google
http://dbappl.cs.utwente.nl/pftijah/
PF/Tijah | Main / Home Page
PF/Tijah
Search: Go
University of Twente
The Netherlands
MONETDB
SOURCEFORGE.net
Done

PF/Tijah Home
Features and Goals
Download
Publications
Links
Contact

Documentation
Running the Server
Getting Started
Show Cases
Install from CVS
Quick Reference

Main /

Home Page

PF/Tijah (Pathfinder/Tijah, pronounce as "*Pee Ef Teeja*") is a flexible open source text search system developed at the University of Twente in cooperation with CWI Amsterdam, the University of Tübingen, and the University of Konstanz. The system is integrated in the Pathfinder XQuery compiler and can be downloaded as part of the MonetDB/XQuery database system.

News

- 2-3 February 2009: Pathfinder meeting in Schloss Dagstuhl.
- 8 January 2009: Tristan Pothoven and Marijn van Vliet developed a Digital Museum of Information Retrieval Research for accessing the old IR literature using PF/Tijah at http://www.sigir.org/museum
- 29 November 2008: New stable release, PF/Tijah 0.9.0 as part of Pathfinder 0.26.0.
- 13 November 2008: We added the code of a small project that was done by Douwe van der Meij for last year's Information Retrieval course, see ElectionsDemo.
- 30 June 2008: Another stable release. This version supports the "algebra version" of Pathfinder Download PF/Tijah
- 17 June 2008: To facilitate topic development for the Entity Ranking track, we developed a simple but effective INEX entity ranking demo. The demo searches in about 4.5 GB of English Wikipedia articles. It is

References

- Wouter Alink. XIRAF: An XML information retrieval approach to digital forensics. *Master’s thesis, University of Twente*, 2005.
- Forbes Burkowski. Retrieval activities in a database consisting of heterogeneous collections of structured text. In *Proceedings of the 15th ACM SIGIR*, pages 112–124, 1992.
- Charles Clarke, Gordon Cormack, and Forbes Burkowski. An algebra for structured text search and a framework for its implementation. The Computer Journal 38:43–56, 1995.
- Djoerd Hiemstra, Henning Rode, Roel van Os and Jan Flokstra, “PFTijah: text search in an XML database system’’, *Workshop on Open Source Information Retrieval (OSIR)*, 2006.
- G.H. Gonnet and F.W. Tompa. Mind your grammar: a new approach to modelling text. In *Proceedings of 13th VLDB*, 1987
- Jani Jaakkola and Pekka Kilpeläinen. Nested text-region algebra. *Technical report, University of Helsinki*, 1999.
- Richard O'Keefe and Andrew Trotman. The Simplest Query Language That Could Possibly Work. In *INEX 2003 Workshop Proceedings*

Acknowledgements

Jan
Flokstra

Henning
Rode

MultimediaN