Queries: INEX 2003 working group report

Börkur Sigurbjörnsson
Language & Inference Technology Group
University of Amsterdam
Amsterdam, The Netherlands
borkur@science.uva.nl

Andrew Trotman
Department of Computer Science
University of Otago
Dunedin, New Zealand
andrew@cs.otago.ac.nz

1. INTRODUCTION

This paper summarizes the discussion of the queries working group at INEX 2003. The group discussed both Content-Only (CO) and Content-And-Structure (CAS) queries. Discussion was however mainly on CAS query syntax, CAS target elements and future CAS data types. The queries working group consisted of: Holger Flörke, Norbert Fuhr, Kenji Hatano, Börkur Sigurbjörnsson, Andrew Trotman, Masahiro Watanabe

Content Only Topics

There was little discussion on CO topics in the working group. This is to be interpreted as a support for leaving the CO topic format unchanged for at least next year.

There was a brief discussion about *query classification*, similar to the classification in [2]. It was considered useful to create a post-hoc classification of the CO queries. Participating groups could then compare their systems performance w.r.t. different types of queries.

Content And Structure Topics

The main discussion was about the complexity of the INEX 2003 CAS queries. It seems that people find it difficult to formulate the XPath-like expressions of the topic title. In the initially distributed (yet reviewed) set of CAS queries, 63% of the queries turned out to be in error [4]. This is in line with research that shows that users have great difficulty with boolean queries, both in databases and information retrieval [3]. Note however that the INEX topics were created by experienced IR researchers. In view of the high error rate there was discussion about syntax clarification, expressiveness restrictions and even a new syntax [4].

The possibility of creating a query generation tool was briefly discussed. The idea was that this tool would help to eliminate mistakes caused by a cumbersome syntax. No details were discussed about the precise functionality of the tool.

There was little discussion about the VCAS task. It is problematic to tell if the CAS queries are suitable for the VCAS task, since the evaluation method for VCAS has not been developed. That is, it is not clear what the task actually is.

In the remainder of this paper we will discuss the two issues which got the most attention from the working group; natural information need in CAS topics; and CAS topic format for INEX 2004.

2. INFORMATION NEED (CAS)

On top of difficulties with the topic syntax, there was also discussion about the difficulty of expressing a natural information need with the current collection. It was questioned whether topic authors add structural constraints because they think it is useful or whether they do it only because they need to write a structured query. The current collection is not very semantically rich and therefore there are limited opportunities for introducing interesting structural constraints.

The working group discussed separately the natural-ness of *target elements* and *structural conditions*.

2.1 Target elements

The working group tried to identify natural target elements for the INEX 2003 collection. The group could identify a few semantically different types of targets.

Textual elements

Textual elements are elements such as sections and paragraphs (`<sec>`, `<ss1>`, `<p>` etc.). It is not obvious which textual tag-name is the most appropriate for a particular query. The question of relevance is more based on the text than the tag-name. It is therefore probably best to leave this problem to the retrieval systems to solve.

Vitaes

Vitaes (`<vt>`) are indeed textual elements, but their semantics is different from the layout semantics of the textual tags in the previous section.

Abstracts

Abstracts (`<abs>`) are also textual elements with a slightly different semantics, since they contain a condensed description of the content of an article, and no detail information.

Bibliographical entries

Bibliographic entries are a different class of answers, since they contain only references to publications, but no real "content" like the textual elements. They represent information needs such as "find references to papers about compression" or "give me all bibliographic details of publications cited within papers about compression".

Note that for example queries such as

```
//article[about(.,'neural networks')]//fm//au
```

which says something like "give me authors of articles about neural networks" are not considered interesting. From an IR perspective this query is equivalent to the query "give me articles about neural networks". The problem of extracting the authors is trivial. Therefore author names is not considered here as a natural target.

The above list is based on discussion in the working group and it is not necessary complete. If topic authors find other natural target elements they are encouraged to use them.

2.2 Structural conditions

The working group distinguished three natural types of structural conditions.

Co-occurrences

We want certain concepts to be covered in the same unit. Say, for example we would like to retrieve documents that discuss the use of handheld computers in health care. We would like to minimize the change of getting documents that discuss handheld computers and health care separately. We could try to express this in a query that asks for articles were handheld computers and health care are discussed in the same section.

```
//article[about(.//sec,'handheld computers health
care')]
```

Note that since we are doing IR, we do not enforce term occurrence restrictions. By co-occurrences we are referring to the co-occurrence of concepts but not terms.

Data-types

Data-types are interesting for retrieval in structured documents. For this particular collection they are of limited use. They should however be considered in retrieval from semantically richer collections which contain not only layout semantics. Examples are markup for chemical processes, financial market developments and geographical locations.

Roles

We want to restrict our attention to XML elements that represent a certain role; such as article author, author affiliation, etc. For example if we want articles authored by Bruce Croft:

```
//article[about(.//fm//au,'Bruce Croft')]
```

Similarly if we want articles that cite Bruce Croft:

```
//article[about(.//bb//au,'Bruce Croft')]
```

We could also restrict our attention to articles where an author is affiliated in California:

```
//article[about(.//fm//au//aff,'California')]
```

This list of natural constraints must be viewed in the context of the current collection. Different collections have different information needs. For collections that have a larger variety in tag-names it is probably easier to formulate natural structural queries.

2.3 Separation of constraints and targets

It was discussed whether the structural constraints and targets needed to be expressed in the same expression. More precisely the question was whether we should go back to the INEX 2002 notation. The main reason behind abandoning the INEX 2002 notation, was that the semantics of that notation was unclear.

Consider for example the query

```
//sec[about(.,'solar powered robots')
and about(.//fig,'robot on mars')]
```

Where we want the retrieved sections to contain figures. Note however, that this is perhaps not a good example for the current LaTeX–originated collection, since authors often use tricks to include figures.

3. QUERY LANGUAGE FOR INEX 2004

This section will report on the discussion within the working group about requirements for a query language. We will then outline the syntax and semantics of a query language that is currently being constructed as a future language for INEX.

3.1 Requirements

The existing syntax of CO proved adequate. Any changes must maintain compatibility with the existing CO topics.

As a query language for CAS titles, the group considered an extension of a subset of XPath. The idea is to take the current syntax extension of XPath, used at INEX 2003, but restrict the usage to an "IR minimum" as described in [4]. This restriction in functionality supports all the important features used in previous workshops. Some queries are known to contain deprecated features and are excluded from this compatibility requirement.

There already exist two data types, numeric and string. This is anticipated to expand in the future to include names, units of measure, and even geographic locations. The language must be extensible to include these at a future date.

Tag instancing is to be deprecated. Restricting a search to a first paragraph (p[1]) was considered unnecessary and unlikely to be used. Query 13 already uses this feature, but this query was considered contrived. Furthermore no relevance assessments are available for this query.

The use of XPath axis, the plethora of XPath syntax for discussing paths, is to be limited to the descendant axis. In particular, the child axis is to be outlawed. None of the queries used so-far, relied on the usage of the child axis. The child axis can be added at any time if a future collection calls for such information need. Path filtering is to remain. Application of multiple filters is to remain.

Use of the (not)-equal operator is to be deprecated for the string data-type. All textual queries are to be expressed in terms of the about predicate. For arithmetic qualification the operators are to be limited to >, <, =, >=, <=.

The semantics must be interpretable vaguely. The XPath semantics are clearly defined making it a database language. For INEX, an IR language is needed, one in which the semantics can be determined by the retrieval engine. In particular, the meaning of the Boolean operators "AND" and "OR" is to become loose and vague.

Multiple target elements is to be deprecated. Queries can specify only one target element. Queries with unspecified target elements are to be added. In these queries the retrieval engine is to choose the most appropriate target element.

Equivalence tags are to remain, but are beyond the scope of the query language.

3.2 Syntax and Semantics

Work is going on to create a detailed description of a query language for INEX 2004. We will mention the most important features here but the full details are beyond the scope of this paper and should be covered in the topic development guidelines.

For the CO topics there is no change from last year.

For the CAS topics we will only discuss the topic title. Other fields do not change between years. The CAS title queries can take two forms

```
//A[B]
//A[B]//C[D]
```

where A and C are path specifications but B and D are filters. To provide backward compatibility we should also consider the form

```
//A[B]//C
```

but as mentioned in a previous section, the added value of this type of topics for an IR test collection is none. The projection `//C` is trivial.

Paths

A path through the XML tree is specified as a sequence of nodes. The only relationship between nodes in a path is descendant. Child relationships are not supported. The wildcard '*' can be used as to refer to a unspecified type of target element. There is a question whether there is a need for including attributes for this collection. There is no (yet assessed) topic that uses attributes.

Strict interpretation: "//A" means any A tag in the tree. "//A//B" means any B descendant of an A tag in the tree. "//@C" means the C attribute of any tag. "//A//@C" means any C attribute anywhere in the tree beneath an A tag in the tree. "//A//*" is any descendant of A. "//*" is any descendant of the root, which also means any tag in the tree.

Loose interpretation: There is likely to be relevant information in the document in places not specified in a user query. The path specifications should therefore be considered hints as to where to look.

Filters

We support one string predicate and several numerical comparisons within the filters.

We use the `about(path,text)` string predicate used in INEX 2003. The textual part of this predicate should always be interpreted in a vague fashion. That is, the validity of the predicate will always need to be done by a human assessor. For example, the query

```
//article[about(.//p, '"information retrieval"')]
```

is strictly interpreted as "Return article tags for only those documents that contain a p tag whose content is about information retrieval". It is loosely interpreted as "What I want is most likely a whole article that discusses information retrieval in a p tag. Relevant results are not limited to this, but I'm pretty sure it'll help you find what I want."

For numeric values we support the operator <,>,=,>= and <=. As with string qualification, this is specified with a relative path. As an example. To "strictly" retrieve article tags from documents published during 2001 we write

```
//article[.//pdt//yr = 2001]
```

this query could equally be specified using string qualification as

```
//article[about(.//pdt//yr, '2001')]
```

In this example, a loose interpretation could be to ignore the qualification or to say that the article should be published around 2001-ish.

The above search predicates and comparison operators can be combined by the Boolean operators AND and OR. Also brackets can be used. Strict interpretation would be that the Boolean operators are strictly interpreted. Loose interpretation: AND is interpreted as ANDish, OR as ORish. The query contains the Boolean operators as hints on how to resolve the information need.

Examples

Examples of some CAS queries are given here along with strict interpretations. Loose interpretation of each is the same "I'm sure this'll help find what I want".

```
//sec[about(., 'mobile electronic payment system')]
```

Return sec tags where the sec tag mentions mobile electronic payment systems.

```
//*[about(., 'singular value decomposition')]
```

Return elements about singular value decomposition. This is a combination CAS-CO query where the retrieval engine must deduce the most appropriate element to return.

```
//article[.//fm//yr >= 1998]//sec[about(.,
'"virtual reality"')]
```

Return sec elements of documents where the yr tag under the fm tag under the article tag is numerically greater than or equal to 1998, and where a sec tag discusses 'virtual reality'.

```
//article[(.//fm//yr = 2000 OR .//fm//yr = 1999)
AND about(., '"intelligent transportation system"')]
//sec[about(., 'automation +vehicle')]
```

Return sec elements about vehicle automation from documents published in 1999 or 2000 that are about intelligent transportation systems.

We are currently working on a more detailed description of the syntax and semantics of the future INEX query language.

4. REFERENCES

[1] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley-Longman, 1999.

[2] K. Hatano, H. Kinutani, M. Watanabe, Y. Mori, M. Yoshikawa, and S. Uemura. An evaluation of inex 2003 relevance assessments. In *INEX 2003 Workshop Proceedings*, pages 25–32, 2003.

[3] M. Hearst. *User Interfaces and Visualization*, chapter 10. In [1], 1999.

[4] R. A. O'Keefe and A. Trotman. The simplest query language that could possibly work. In *INEX 2003 Workshop Proceedings*, pages 117–124, 2003.