CSIRO's participation in INEX 2006

Alexander Krumpholz and David Hawking

CSIRO ICT Centre, Canberra, Australia
{Alexander.Krumpholz, David.Hawking}@csiro.au

Abstract. In this year's INEX[1] participation, CSIRO participated in the Ad-hoc Track, contributing to three of the four given tasks, namely the Thorough Task, the Focussed Task and the Best in Context Task.
In order to use CSIRO's plain text search engine PADRE[2] we had to preprocess the collection in a similar manner to the approach taken by our group in 2002[3]. We split the documents in subdocuments according to the elements that we need to retrieve and indexed the files with PADRE.
In a first step we extracted query elements from the INEX topics. Then the query processor generated PADRE queries and post-processed the results according to specifications for each run.

1 Introduction

CSIRO participated in the inaugural INEX round in 2002, but not in subsequent years. This years involvement in INEX had mainly two motivations. First, to explore the new Wikipedia collection, second and to see how the naive splitting approach compares to the new XML retrieval engines and second to have a baseline system for comparison with future approaches.

PADRE, CSIRO's text search engine implements a slightly modified Okapi BM25 algorithm. In order for PADRE to retrieve sub-document parts like XML elements, the original XML files had to be split into smaller documents according to the XML elements relevant for retrieval.

PADRE has the ability to index metadata using metadata classes, which are represented by a single character. Default metadata classes are used for HTML to map the title, author and date to different classes. This way a PADRE query can restrict the search for keywords to specific elements. For example the query "t:architecture" would only return pages having a title element matching the term "architecture". In addition to the default metadata classes defined, PADRE allows the user to specify project specific configuration files containing the mapping of XML elements to metadata classes. The xml.cfg mapping file used for this experiment is shown in Listing 1.1.

2 Approaches to INEX Ad-hoc Tasks

This year the INEX community defined four challenges for the Ad-hoc track:

Thorough Task : The goal of the Thorough Task is to retrieve a ranked list of elements over the whole collection.

Focused Task : The Focused Task also asks for elements ranked over the whole collection, but does not allow overlaps. Overlaps occur when multiple retrieved search results contain identical XML elements. This happens when a sub element or super element of an already returned retrieval result ar being returned.

All In Context Task The All In Context task aims for getting a list of ranked documents including the relevant, non-overlapping elements within the document.

Best In Context Task The goal is to return a list of Ranked documents with the best entry point for a reader.

Retrieval of structured data is quite related to database query languages and requires selection and projection operations.

A summary of the different runs is shown in Table 1.

The selection is achieved using the PADRE retrieval engine. In a pre-processing step we converted the NEXI queries into different PADRE queries which could be used by the Query Processor. For the two main query types i.e. Context and Structure (CAS) and Context Only (CO) the PADRE version of the NEXI query could be used. However, the keywords in the CAS and the CO topics are not identical (See Table 4). In order to explore the effect of this difference a third query has been constructed by removing the structural hints from each CAS topic.

The projection aspect has been addressed in a post-processing phase. Using PADRE and the collection of element level sub documents, we based our runs on a number of considerations: The standard PADRE version based on the Okapi BM25 algorithm delivers exactly what the Thorough Task (A) requires. The simplest possible way to achieve the Focussed Task (B) would be to run the same query used for case (A), while suppressing overlaps by skipping results which are in fact descendants (subelements) or ancestors (superelements) of a previously returned result. In order to generate a baseline for the Best In Context Task we took the simple assumption that Wikipedia articles are covering very specific topics and that the article itself would be a reasonable entry point for a user.

The following eight runs were submitted:

Run name	Selection based on	Projection based on
CSIRO-CAS1-A	CAS-Title	n/a
CSIRO-CAS2-A	CAS-Title	query specified elements
CSIRO-CO1-A	CO-Title	n/a
CSIRO-CO1-B	CO-Title	suppression of overlapping results
CSIRO-CO1-D	CO-Title	<article> elements
CSIRO-CO2-A	CAS-Title without structure	n/a
CSIRO-CO2-B	CAS-Title without structure	suppression of overlapping results
CSIRO-CO2-D	CAS-Title without structure	<article> elements

Table 1. CSIRO's runs

3 Architecture

This section describes the architecture used

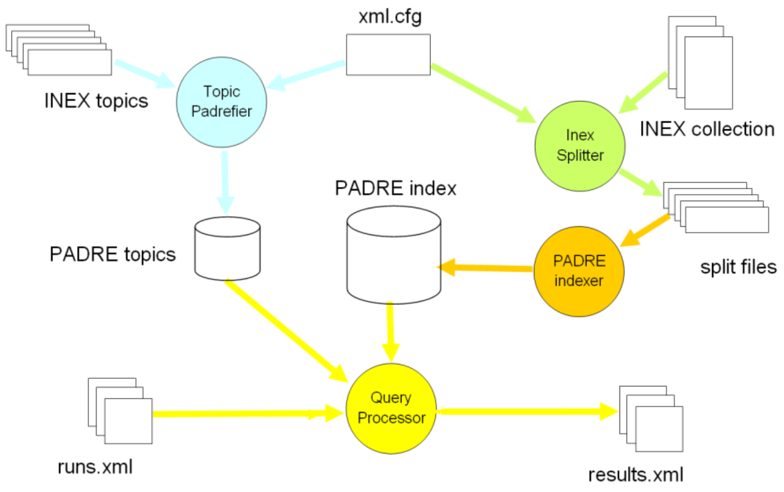

Fig. 1. Architecture

3.1 Defining metadata classes

The xml.cfg specifies the elements indexed and the PADRE metadata classes used to represent them. The DOC element is an artificial element introduced to create a collection of subdocuments extracted from the original XML file. (see section 3.2)

```
document,//DOC
a,1,,//DOC/article
b,1,,/body
e,1,,/caption
f,1,,/p
g,1,,/figure
j,1,,/title
l,1,,/table
n,1,,/template
o,1,,/section
p,1,,/template@name
q,1,,/collectionlink
r,1,,//DOC/article/name
w,1,,/link
x,0,,/DOCNO
```

Listing 1.1. xml.cfg

3.2 Splitting documents

The original XML documents needed to be split into subdocuments in order to exploit PADRE's document retrieval capability to actually retrieve subdocuments. We split each XML document of the collection into subdocuments by extracting all elements matching one of the following XPath expressions and storing them in a new XML cotainer document:

– /article
– //section
– //p
– //template

Error1: Sub-elements are not indexed with the super-element, i.e. metadata classes are disjunct.

3.3 Transforming NEXI topics

INEX topics are specified using the NEXI query language [4]. NEXI is XPath inspired, but allows the usage of vague selectors. The query `//section[about(.//p, security)]` matches sections that have a descendent element <p> containing the keyword 'security'.

In order to use the PADRE search engine we had to convert the official NEXI queries into PADRE queries. The script TopicPadrefier.rb (see figure 1) extracts the selection and projection component of each original CAS and CO topic, constructs the closest possible PADRE queries and stores them for future use. However, PADRE does not allow the definition of an infinite number of metadata classes and is therefor not expressive enough to correctly build PADRE queries for all possible NEXI queries. Some of the complex relationships of elements can therefor not be expressed in a generic manner.

NEXI	//article//figure[about(., Renaissance painting Italian Flemish -French -German)]
PADRE	g:Renaissance g:painting g:Italian g:Flemish -g:French -g:German

Table 2. NEXI and PADRE query for topic 292

NEXI	//article[about(.,wifi)]//section[about(.,wifi security encryption)]
PADRE	a:wifi o:wifi o:security o:encryption

Table 3. NEXI and PADRE query for topic 293

<title>	proprietary implementation +protocol +wireless +security
<castitle>	//article[about(., wireless)and about(.//p, security)] //link[about(., proprietary +implementation)]

Table 4. A <title> and <castitle> element of topic 349

3.4 Running INEX topics

The result for each run has to be submitted as an XML file matching a DTD specified by the organizers. Since the Query processor needed processing instructions for each run, an extended version of the result DTD has been used as input for the Query processor. The Query processor extracts the processing instructions from the run specification file and replaces it with the result for the run. This way the run specification is automatically well documented.

```
<inex-submission participant-id=”22” run-id=”CSIRO-CO1-B”
task=”Focused␣query=”automatic”>
␣␣<topic-fields␣title=”yes”␣castitle=”no”␣description=”no”␣narrative=”no”␣ontopic_keywords=”no”/>
␣␣<description>Using␣the␣title␣as␣a␣query␣to␣padre,␣but␣suppressing␣overlapping␣results
␣␣</description>
␣␣<processing-instructions
␣␣␣␣element-restriction=”None”
␣␣␣␣suppress=”Overlap”
␣␣␣␣padre-blocksize=”1600”
␣␣>
␣␣␣␣<query>query.padre_title.uniq.join('␣')
␣␣␣␣</query>
␣␣</processing-instructions>
␣␣<collections>
␣␣␣␣<collection>wikipedia</collection>
```

```
  </collections>
</inex-submission>
```

Listing 1.2. XML.cfg

3.5 Post-processing results

The results have to be post-processed to meet the given tasks: suppressing multiple documents or limit to element types in question or articles. For the Best In Context task all results other than articles have been suppressed in the runs CSIRO-CO1-D and CSIRO-CO2-D. For the Content and Structure task we suppressed all elements different than the structural request extracted from the query for the run CSIRO-CAS2-A, while we did not modify the result from PADRE for the runs CSIRO-CAS1-A, CSIRO-CO1-A, and CSIRO-CO2-A. For the runs CSIRO-CO1-B and CSIRO-CO2-B we matched each result with the results already delivered to filter out ascended or descended elements.

4 Results

The following table shows our results in respect to the other participant's submissions. Generally we made it into the middle field, with the following interesting observations:

The metrics [5] used for the Thorough Task is ep-gr.

thorough		
Metric:ep-gr Quantization: gen,Overlap=off		
S.No	RunId	MAep
1		0.0709
33	CSIRO_CO2_A	0.0320
36	CSIRO_CO1_A	0.0313
78	CSIRO_CAS1_A	0.0122
80	CSIRO_CAS2_A	0.0111
106		0.0000

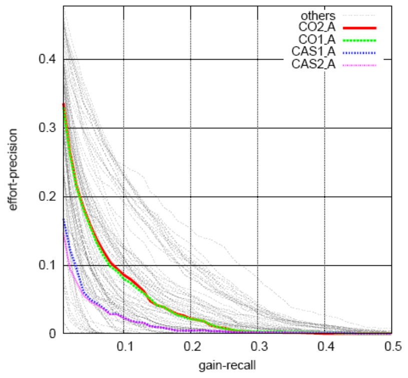

Fig. 2. Result Thorough

The tables list the runs submitted by CSIRO as well as the best and worst result for comparison. Notice, that the largest sample number at the bottom always shows the number of runs submitted for that task.

All results show the two similar approaches almost side by side, indistinguishable from each other in the small graphs attached. Only Figure 2 contains four runs by our team, on pair at position 33 and 36, the second pair at 78 and 80. We could not find a doubtless conclusion for the reason for the Content Only runs to beat the Content And Structure runs by far. This is especially interesting since we would expect the additional information coming from the structure to improve the retrieval quality.

The collocation of the other graphs are due to fact, that the Context Only title and the Context And Structure title usually do not differ a lot.

The runs CAS1-A and CAS2-A share similar qualities as well. Apparently the element PADRE delivers automatically is often the one, that would be specified in a CAS scenario.

focused		
Metric:nxCG Quantization: gen,Overlap=on		
S.No	RunId	nxCG@5
1		0.3944
18	CSIRO_CO1_B	0.3322
21	CSIRO_CO2_B	0.3304
85		0.0000

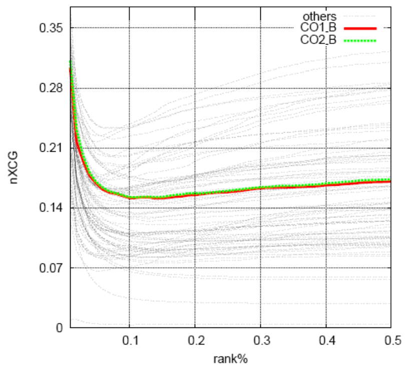

Fig. 3. Result Focused

The Focused task turned out to be the best of our results, even though the approach is quite simple.

Observations: Our structured results are worse than the unstructured ones.

Some graphs.

4.1 Failure/Success Analysis

A first Failure Analysis of the results made us run additional experiments. One potential improvement was the option -vsimple to switch off web-specific features like path length punishment. While a long URL might refer to an unimportant element far down the hierarchy, the same mechanism would rank top elements always higher then their descendants. However implementing this feature reduces our nxCG@5 from 0.3322 (Focused task / run CSIRO-CO1-B) to 0.3261. Some of our files still contained XPath expression, which should not be in the INEX run submissions, namely elements without a position of 1 ($/h3/$ instead of $/h3[1]/$).

BestInContext		
Metric: EPRUM-BEP-Exh-BEPDistance		
S.No	RunId	At A=0.01
1		0.0407
41	CSIRO_CO1_D	0.0166
43	CSIRO_CO2_D	0.0161
77		0.0000

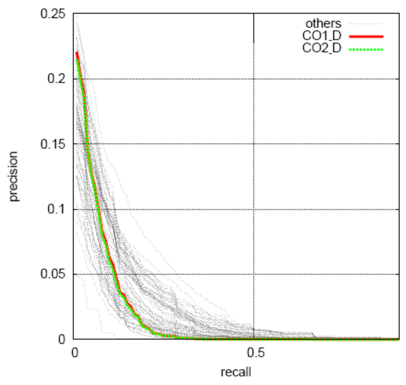

Fig. 4. Result Best in Context

Fixing those problems unfortunately caused a further decrease of the nxCG@5 to 0.3216.

5 Conclusions

Examining the results indicates that the XML retrieval community progressed significantly in recent years and just using a full text search engine to 'mimic' structured retrieval does not deliver competitive results any longer.

In order to get into the game, future work will go into an engine that implements the findings of the INEX community in the past or to start out with one of the established systems and to improve certain aspects of algorithms.

References

1. Infos about INEX
2. Infos about PADRE
3. Infos about CSIRO at INEX 2002
4. Infos about NEXI
5. Infos about Metrics