

Finding the Best Entry Point

A thesis
submitted to the faculty of the graduate school
of the University of Minnesota
by

Sarika Kishor Mehta

In partial fulfillment of the requirements
for the degree of
Masters of Science

August, 2008

Department of Computer Science
University of Minnesota Duluth
Duluth, MN 55812
USA

UNIVERSITY OF MINNESOTA

This is to certify that I have examined this copy of a master's thesis by

Sarika Kishor Mehta

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee has been made.

Dr. Carolyn Crouch

Name of the Faculty Adviser

Signature of the Faculty Adviser

Date

GRADUATE SCHOOL

© Sarika Kishor Mehta, 2008

Acknowledgements

I would like to thank Dr. Carolyn Crouch for giving me an opportunity to work with her and guide me throughout my thesis work. Her immense knowledge in the field of information retrieval has been an inspiration for me.

I would also like to thank Dr. Donald Crouch for his support during my work. I would like to thank Dr. Robert McFarland for his valuable inputs.

I would like to thank the staff of the Department of Computer Science at UMD, especially Jim Luttinen, Lori Lucia and Linda Meek. I would also like to thank my colleagues Salil Bapat and Darshan Paranjape for their suggestions and regular help. Thanks to Pete Willemsen for his help during my summer work.

Special thanks to Pavan Poluri for his help in INEX result submissions and running experiments. I would also like to thank Dinesh Bhirud, Varun Sudhakar, Chaitanya Polumetla for their assistance during my work.

Finally, I would like to thank my father, Mr. Kishor Mehta and rest of my family, all my professors, and friends in Duluth who have been a constant source of encouragement during my two years in Duluth.

Abstract

Information retrieval (IR) is the technique of storing document-related data efficiently and extracting relevant information from it. The desired information may be a document, some portion of a document, metadata, information in a stand-alone or networked database or information from the World Wide Web. Most of the queries fired over the web require a text response. A major portion of data on the web is in XML format. Relaxing rules for web document development to encourage user participation has led to more XML-naïve users' modifying and maintaining large amounts of web data. The result is an increasing amounts of untagged text in XML documents (also called semi-structured information). Retrieving relevant data from these semi-structured documents now poses an interesting challenge for researchers.

This thesis focuses on finding the Best Entry Point (BEP) for relevant documents. This is in conformance with the INitiative for the Evaluation of XML Retrieval (INEX) Adhoc track *Best-in-context* task. The BEP is defined as the place to start reading a document to reference the information specified by the query. Based on the BEP analysis from the INEX 2006 collection [12] and analysis performed on the INEX 2007 relevance assessments, several strategies have been designed for the *Best-in-context* task. This thesis discusses the results obtained by these techniques.

Table of Contents

1. Introduction	01
2. Background	03
2.1 Vector Space Model.....	03
2.2 Smart.....	03
2.3 Flexible Retrieval.....	05
2.4 Overview of All-element retrieval.....	06
2.4.1 Parsing.....	06
2.4.2 Indexing.....	08
2.4.3 Slope and Pivot Computation.....	08
2.4.5 Evaluation.....	10
3. Experimental Setup	14
3.1 Document Collection.....	14
3.2 Query Collection.....	16
3.3 Retrieval Tasks and Evaluation Metrics.....	18
3.3.1 Retrieval Tasks 2007 and 2008.....	18
3.3.2 Relevance Assessment.....	19
3.3.3 BEP Metric.....	19
4. Parsing and Relevance Assessment Analysis	23
4.1 Parsing the Wikipedia Collection.....	23
4.1.1 Article Parse.....	24
4.1.2 Section Parse.....	25
4.1.3 Para Parse.....	26
4.1.4 Para+mt Parse.....	27
4.2 Relevance Assessments 2007 v4 Analysis.....	28
5. Best-In-Context Experiments	31
5.1 BEP based on Correlation Score.....	32
5.2 XPath Collection.....	36
5.3 Analysis.....	40

6. Conclusions and Future Work	42
6.1 Conclusions.....	42
6.2 Future Work.....	43
References	44
Appendix A.....	46
Appendix B.....	56
Appendix C.....	59

List of Figures

Figure 1: Lnu weighting scheme formula.....	04
Figure 2: Ltu weighting scheme.....	05
Figure 3: Flowchart depicting the procedure for all-element retrieval.....	09
Figure 4: Format of the EvalJ output for the <i>focused</i> task.....	11
Figure 5: Format of the EvalJ output for the <i>relevant-in-context</i> task.....	12
Figure 6: Format of the EvalJ output for the <i>best -in-context</i> task.....	13
Figure 7: Difference in image tags between the collections of 2007 and 2008.....	15
Figure 8: Sample Wikipedia semi-structured XML document.....	16
Figure 9: Sample topic from INEX 2007 v4 topic pool [Topic id = 414].....	17
Figure 10: Sample element from article parse.....	24
Figure 11: Sample element from section parse.....	25
Figure 12: Syntax of special cases in para parsing.....	26
Figure 13: Sample output of para parse.....	27
Figure 14: Sample output of para+mt parse.	27
Figure 15: Sample XPath file for document id 1000168.xml.....	37

List of Tables

Table 1: Number and type of elements indexed the INEX 2007 document collection....	06
Table 2: Tags identified as terminal nodes for Wikipedia 2007 and 2008 document collection.....	07
Table 3: Slope and Pivots values for INEX 2008 collection.....	08
Table 4: Comparison between all 22 sections of the 2007 and 2008 wikipedia collection.....	15
Table 5: Relevance assessment 2007 V4 analysis for Best Entry Point (BEP) elements..	29
Table 6: Summary of BEP analysis from relevance assessments 2007 v4.....	30
Table 7: Highest correlating element (no restriction on type of BEP); Input: RIC; Index: New tag set	32
Table 8: Highest correlating element (no restriction on type of BEP); Input: RIC; Index: New tag set revised.....	33
Table 9: Highest correlating element (no restriction on type of BEP); Input: Flex; Index: New tag set	33
Table 10: Highest correlating element (no restriction on type of BEP); Input: Flex; Index: New tag set revised.....	34
Table 11: Highest correlating element; Set: New tag set; Input: RIC; Index: New tag set	35
Table 12: Highest correlating element; Set: New tag set revised; Input: RIC; Index: New tag set revised.....	35
Table 13: Highest correlating element; Condition: BEP € New tag set; Input: Flex; Index: New tag set	36
Table 14: Highest correlating element; Condition: BEP € New tag set revised; Input: Flex; Index: New tag set revised.....	36
Table 15: Highest physical position (no restriction on type of BEP); Input: RIC; Index: New tag set.....	38
Table 16: Highest physical position (no restriction on type of BEP); Input: RIC; Index: New tag set revised.....	39

Table 17: Highest physical position (no restriction on type of BEP); Input: Flex; Index: New tag set.....	39
Table 18: Highest physical position; Condition: BEP € New tag set; Input: RIC; Index: New tag set.....	40
Table 19: Highest physical position; Condition: BEP € New tag set revised; Input: RIC; Index: New tag set revised.....	40
Table 20: Relevance Assessment 2007 V4 Analysis: BEP per topic.....	59

1. Introduction

Information retrieval (IR) is the technique of storing document-related data efficiently and extracting relevant information from it. The desired information may be a document, some portion of a document, metadata, information in a stand-alone or networked database or information from the World Wide Web. Most of the queries fired over the web require a text response.

With the increasing growth of web-based data, the user needs to find answers to his/her queries efficiently. Text on the web resides in multiple formats, including unstructured, semi-structured and structured forms. In February 1998, the World Wide Web Consortium [18] published eXtensible Markup Language (XML) [19]. A major portion of data on the web is in XML format. Relaxing rules for web document development to encourage more user participation has led to more XML-naïve users' modifying and maintaining large amounts of web data. The result is an increasing amounts of untagged text in the XML documents (also called semi-structured information). Retrieving relevant data from these semi-structured documents now poses an interesting challenge for researchers.

Early IR systems dealt with pure unstructured text and returned references to documents in response to a query. It was then up to the user to search for details. Current web-based systems focus on handling structured text (our system for flexible retrieval, Flex [10], was developed specifically to deal with structured text). Such systems aim to capture very specific information by returning small portions of text instead of the entire document. Semi-structured documents raise new issues and systems must be adapted to ensure compatibility with this new domain. Flex was modified specifically to deal with

the presence of untagged text in documents [1, 9, 12].

The INitiative for the Evaluation of XML Retrieval (INEX) is a research initiative that focuses on aspects of XML retrieval [7]. The University of Minnesota Duluth (UMD) IR group participates in the INEX competitions. There were 105 participants in 2007 [11]. INEX provides the document collection and evaluation measures for the comparison of different systems. The aim is to discover effective ways of managing and retrieving XML data.

The way in which XML documents are represented impacts the effectiveness of retrieval, so it is essential to choose the proper representation. Our basic retrieval system, Smart [16], is based on Vector Space Model (VSM) [16]. The collection on which the experiments are performed is the Wikipedia collection [5], provided by INEX. Experimental results are evaluated using the INEX evaluation software, *EvalJ* [3]. The main focus of this thesis is to improve the *Best-in-context* results by using parsing techniques based on relevance assessments and 2007 evaluation measures to achieve a better performance.

Chapter 1 is a brief introduction to our work in semi-structured XML retrieval. Chapter 2 focuses on Smart and Flex. Chapter 3 gives an overview of INEX tasks, evaluation measures and document collection. Chapter 4 discusses parsing challenges and the relevance assessments from 2006 [8] and 2007. Chapter 5 describes the experiments performed for the *Best-in-context* task, results and an analysis of the results. Chapter 6 draws conclusions and discusses future work.

2. Background

The process of all-element retrieval consists of parsing, indexing, Lnu-ltu [17] weighting, retrieval and evaluation. Adapting this model to semi-structured documents involves marking untagged elements with the *mt* tag (short for magic text). The entire process is similar to the pure XML element retrieval process except that the *mt* tags have to be removed before the evaluation phase. This retains the untagged text so that this information can be properly indexed [9, 12]. The details of parsing techniques are discussed in Chapter 4. Smart (discussed in Section 2.2) is used for indexing and all-element retrieval. The output from Smart is then input to Flex (covered briefly in Section 2.3). Flex dynamically builds the element vectors and calculates the similarity between the element and query vectors. The result of this process is a ranked list of elements.

2.1 Vector Space Model

The Vector Space Model [16] represents text documents as vectors of identifiers. Each dimension corresponds to a separate term. If a term occurs in the document, its value in the vector is non-zero. A term is weighted using a function of its frequency within the document. The relationship between a query and a document is determined by the distance between the two vectors in vector space. The Vector Space Model is the basic model used by our retrieval engine, Smart 13.0.

2.2 Smart

We use the Smart 13.0 retrieval engine for all retrieval functions. Smart was

developed at Cornell University under the supervision of Gerard Salton by Chris Buckley and others [16]. This engine is based on the VSM.

Raw XML elements, obtained after parsing and conversion to Smart input format, are fed to Smart. A unique identifier is assigned to each index term. This mapping of the unique identifier and the XPath of the element to its physical location is stored in the *textloc* file, which gives the location of the text pointed to by the identifier.

Once all elements have been indexed using Smart, the resulting vectors can be modified via a term weighting scheme. We use Singhal's Lnu-ltu weighting scheme [17]. The formula for the Lnu element weighting scheme is given in Figure 1.

$$\frac{\frac{1 + \log(\text{tf})}{1 + \log(\text{average tf})}}{(1 - \text{slope}) + \text{slope} * (\# \text{ unique terms}) / \text{pivot}}$$

where, tf is term frequency (as in the *nnn* vector)
 average tf is the average term frequency of all terms in this vector
 # unique terms is the number of distinct terms in this vector
 slope & pivot are empirically determined constants.

Figure 1: Lnu weighting scheme formula [17]

The Lnu-ltu weighting scheme uses pivoted length normalization [17]. The formula for the ltu query weighting scheme is shown in Figure 2. The variables N and n_k represent

global statistics and can be calculated using the method described by Ganapathibhotla in [4]. Similarity between the Lnu-weighted vector and the ltu-weighted query is measured using the inner product.

$$\frac{(1 + \log(\text{tf})) * \log(N/n_k)}{(1 - \text{slope}) + \text{slope} * (\# \text{ unique terms}) / \text{pivot}}$$

where tf is the term frequency
N is the collection size
n_k is the number of documents that contain this term
slope & pivot are empirically determined constants
unique terms is the number of distinct terms in this vector

Figure 2: ltu weighting scheme

2.3 Flexible Retrieval

The process of dynamically retrieving the elements from a set of documents is referred to as flexible retrieval. With respect to a query, flexible retrieval returns documents based on highly correlating elements. It is designed to handle both structured and semi-structured documents.

Flexible retrieval uses a base case for evaluation. The baseline is provided by all-element retrieval. In this approach, an index of all the elements is created. The disadvantage of this approach is that multiple copies of data are created due to the overlapping of elements. It also uses considerable disk space due to large file sizes.

We use our flexible retrieval system, Flex, in our work to generate *Best-in-context* results. A complete overview of Flex procedures is discussed in [2, 13]. Leaf nodes, required in Flex operation, are listed in Section 2.4.1. Table 1 shows the numbers of elements of interest.

Type of terminal element	Count of the terminal element
sections	1,602,117
paras (all terminal nodes)	4,936,686
mt	791,895
paras+mt	5,728,581
number of articles	659,395
all-elements	7,198,199

Table 1: Number and type of elements of interest in the INEX 2007 document collection.

2.4 Overview of All-element retrieval

The following paragraphs outline the process of all-element retrieval. Figure 3 is a flowchart which represents this process.

2.4.1 Parsing

The INEX document collection is downloadable as a tar.gz file from the INEX website [7]. Extraction of this .gz file results in a parent folder which in turn contains 22 sub-folders. Each sub-folder contains approximately 30,000 documents. Each of these documents is an XML document that has to be converted to individual elements via parsing. Three different parses are generated in order to meet the needs of Smart and

Flex. These are the Article parse, Section parse and Para parse. Table 2 identifies two sets of terminal node tags: the 2007 set (the old tag set) and the 2008 set (the new tag set)

Element to index	Terminal tags for year 2007 (old tag set)	Terminal tags for year 2008 (new tag set)
Paragraph	<p>.....</p>	<p>.....</p>
Normal-list	<normallist>.....</normallist>	dropped
Ordered-list	dropped
Unordered-list	dropped
Number-list	<numberlist>.....</numberlist>	dropped
Definition-list	<definitionlist>.....</definitionlist >	dropped
Figure	<figure>.....</figure>	<figure>.....</figure>
Table	<table>.....</table>	dropped
Name	-	<name>.....</name>
emph3	-	<emph3>.....</emph3>
Section	-	-
Body	-	-
Article	-	-

Table 2: Tags identified as terminal nodes for Wikipedia 2007 and 2008 document collections

2.4.2 Indexing

The next step after parsing is indexing using Smart 13.0. The parses convert

specific XML text elements to a form that can be input to Smart. The following indices are generated using Smart:

- **article index**

Input: article parse. Use: article retrieval.

- **paragraph+mt index**

Input: paragraph+mt parse. Use: by flexible retrieval to produce the document trees.

- **all-element index**

Input: article, section and paragraph parses. Use: all-element retrieval.

2.4.3 Slope and Pivot Computation

The computation of slope and pivot is done empirically. Slope and pivot are computed for all indices in order to perform pivoted length normalization. The best value is the one that gives the best results for a task upon INEX evaluation. Table 3 shows the slope and pivot values for various indices. Retrievals are then generated as required by the specific task.

Index	Slope/Pivot for 2007	Slope/Pivot for 2008
All-element	0.12/38	0.12/31
Article	0.04/120	0.04/120
Paragraph + mt	0.12/18	0.12/15

Table 3: Slope and Pivots values for INEX 2008 collection

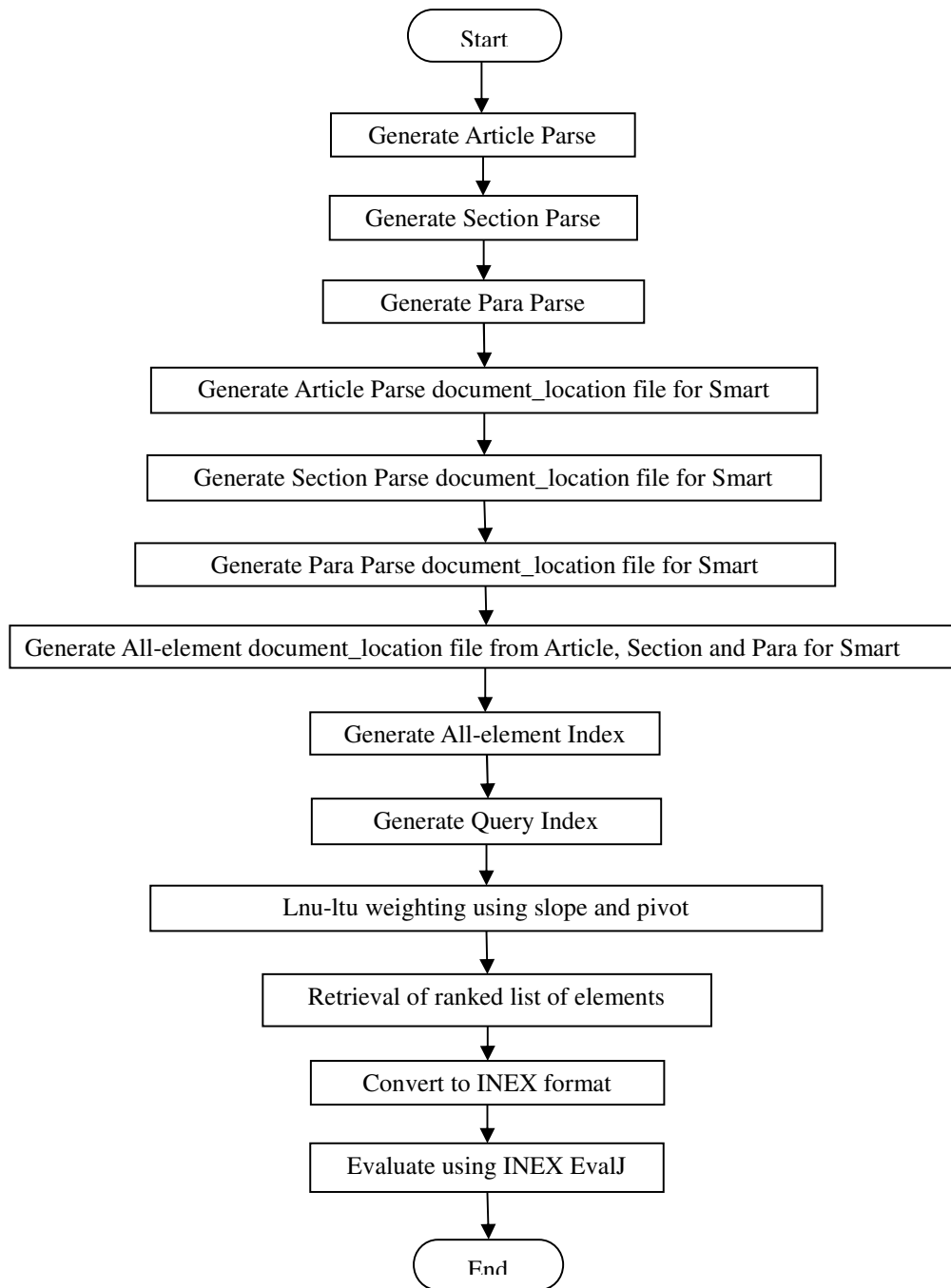


Figure 3: Flowchart depicting the procedure for all-element retrieval

2.4.4 Evaluation

For 2007, the EvalJ package was not available. Instead, a set of Perl scripts was provided by INEX. They required that a 13 GB database be created and certain packages be installed.

The output of Smart is a top-ranked list of elements (*tr_file*). The “*tr.Lnu.txt*” is the Smart “*db_file*” which holds the query id, document id, rank and similarity of all the top-ranked elements. The *tr_file* is converted to Flex format to take advantage of an existing script which performs the conversion to INEX submission format. The Flex file has a query id, XPath and a similarity score for every element. The difference between the *tr_file* and the Flex file is that the Flex file contains the complete XPath rather than the document id. The Flex file is then converted to INEX format. The format of the final result submission file varies depending on the task and the year of the submission. Only after the validation can the file be evaluated using EvalJ.

Figures 4, 5 and 6 show the format of the EvalJ output for each of the three tasks, *Focused*, *Relevant-in-context* and *Best-in-context*, respectively. Our Particular interest lies in Figure 6.

```
<eval run-id="p51_Focused"
file="/smart/08/08_indices/revised_tags/all_el_eval/evaluations/12_31/focused/foc-inex-
12-31.xml">
num_q      all    107
num_ret    all    84677394
num_rel    all    10658789
num_rel_ret all    3181661
iP[0.00]  all    0.533067190690608
iP[0.01]  all    0.470594999001154
iP[0.05]  all    0.342402029749482
iP[0.10]  all    0.261886792698553
MAiP      all    0.0848905400867509
ircl_prn.0.00 all    0.533067190690608
ircl_prn.0.01 all    0.470594999001154
ircl_prn.0.02 all    0.421515294978888
ircl_prn.0.03 all    0.393759802092959
ircl_prn.0.04 all    0.360297695689475
ircl_prn.0.05 all    0.342402029749482
          .
          .
          .
ircl_prn.0.95 all    0.000184416051338857
ircl_prn.0.96 all    0.000184416051338857
ircl_prn.0.97 all    0.000184416051338857
ircl_prn.0.98 all    0.000184416051338857
ircl_prn.0.99 all    0.000184416051338857
ircl_prn.1.00 all    0.000184416051338857
</eval>
```

Figure 4: Format of the EvalJ output for the *Focused* task

```

eval run-id="p51_ric"
file="/smart/para0101/Darshan/2008_FinalExperiments/07/rel_in_context/xml/400/flex_
4000.xml">
num_q      all    107
num_ret    all    27775
num_rel    all    6450
num_rel_ret all    3199
MAgP      all    0.106654155901777
ircl_prn.0.00 all    0.405246736541677
ircl_prn.0.10 all    0.292062101546415
ircl_prn.0.20 all    0.210989280101356
ircl_prn.0.30 all    0.162085745180681
ircl_prn.0.40 all    0.126468218239357
ircl_prn.0.50 all    0.08961277874532
ircl_prn.0.60 all    0.055879009916651
ircl_prn.0.70 all    0.0303763271860859
ircl_prn.0.80 all    0.00692238239765418
ircl_prn.0.90 all    0.000990092869410245
ircl_prn.1.00 all    0.000751183330259998
gP[5]     all    0.230166929659565
gP[10]    all    0.202147525480116
gP[25]    all    0.149754604118329
gP[50]    all    0.110983645659058
gR[5]     all    0.100708496920918
gR[10]    all    0.159607351897153
gR[25]    all    0.253119351384416
gR[50]    all    0.327585898881509
</eval>

```

Figure 5: Format of the EvalJ output for the *Relevant-in-context* task

```

eval run-id="p72_best_in_context"
file="/smart/mehta051/bic/bic_from_ric/bic_07/400/redone/xml/ordered_bic_flex_4000.xml">
num_q      all    107
num_ret    all    36646
num_rel    all    6491
num_rel_ret all    3474
MAgP      all    0.140336106226256
ircl_prn.0.00 all    0.561609508547896
ircl_prn.0.10 all    0.38725249096281
ircl_prn.0.20 all    0.275067493720742
ircl_prn.0.30 all    0.203376147721957
ircl_prn.0.40 all    0.158840621429465
ircl_prn.0.50 all    0.116268534046073
ircl_prn.0.60 all    0.0804998743844671
ircl_prn.0.70 all    0.0424093911140266
ircl_prn.0.80 all    0.0113315135677571
ircl_prn.0.90 all    0.00266888093126575
ircl_prn.1.00 all    0.00225379401637684
gP[5]     all    0.309702803738318
gP[10]    all    0.26844953271028
gP[25]    all    0.199584672897196
gP[50]    all    0.149581495327103
gR[5]     all    0.100094047330219
gR[10]    all    0.158535879269876
gR[25]    all    0.251752501826762
gR[50]    all    0.326311578180041
</eval>

```

Figure 6: Format of the EvalJ output for the *Best -in-context* task

3. Experimental Setup

This chapter describes the document collection used for our experiments. It also describes the retrieval tasks and evaluation metrics used to evaluate our work.

3.1 Document Collection

The INEX 2007 (and 2008) document collection is a set of XML documents which are a part of the Wikipedia corpus [5]. This corpus is composed of eight main collections corresponding to eight different languages. Each collection is encoded in the UTF-8 format. We use the English version of the corpus. The documents of the Wikipedia XML collection are organized in a hierarchy of categories defined by the authors of the articles. For each main collection, there is a set of files describing:

- the hierarchy of categories
- the categories associated with each article
- the names of those categories

The total number of documents in the 2008 collection is 659,388 which uses about 6 GB. One difference between the 2007 and the 2008 collections is that the 2008 collection does not have document '1859874.xml'. The 2008 collection has Image IDs which were not present in the 2007 collection. This difference is shown in Figure 7. The entire Wikipedia collection is divided into 22 parts. Table 4 shows a comparison of all the sections for the 2007 and 2008 collections. The only major difference is in part 0 in which the 2008 collection has an additional 229 documents. A typical Wikipedia semi-structured XML document is shown in Figure 8.

```
<image xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple"
xlink:href="../../../pictures/trianglelogo.jpg" xlink:actuate="onLoad" xlink:show="embed">
```

Figure a: 2007 collection with no image ID

```
<image xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple"
xlink:href="../../../pictures/trianglelogo.jpg" id="-1" xlink:actuate="onLoad" xlink:show="embed">
```

Figure b: 2008 collection with image ID

Figure 7: Difference in image tags between the collections of 2007 and 2008

	07 collection_no of docs	08 collection_no of docs	difference
part - 0	29,771	30,000	229
part - 30000	30,001	30,000	1
part - 60000	30,000	30,000	0
part - 90000	30,000	30,000	0
part - 120000	30,000	30,000	0
part - 150000	30,000	30,000	0
part - 180000	30,000	30,000	0
part - 210000	30,000	30,000	0
part - 240000	30,000	30,000	0
part - 270000	30,000	30,000	0
part - 300000	30,000	30,000	0
part - 330000	30,000	30,000	0
part - 360000	30,000	30,000	0
part - 390000	30,000	30,000	0
part - 420000	30,000	30,000	0
part - 450000	30,000	30,000	0
part - 480000	30,000	30,000	0
part - 510000	30,000	30,000	0
part - 540000	30,000	30,000	0
part - 570000	30,000	30,000	0
part - 600000	30,000	30,000	0
part - 630000	29,387	29,388	1
	659,159	659,388	231

Table 4: Comparison between all 22 sections of the 2007 and 2008 Wikipedia collections

```

<?xml version="1.0" encoding="utf-8"?>
<article>
  <name id="1164753">
    Gauss-Newton algorithm
  </name>
  <conversionwarning>
    0
  </conversionwarning>
  <body>
    The recurrence relation for Newton's method for minimizing a function
    <emph2>
      S
    </emph2>
    is
    <indentation1>
      <math>
        p^{k+1} = p^k -
        <unknownlink src="H (S)(p^k)">
          H (S)(p^k)
        </unknownlink>
        ^{-1}J_S(p^k),
      </math>
    </indentation1>
    where
    <p>
      J_S
    </p>
    and
    <math>
      H(S)
    </math>
    denote the Jacobian and Hessian of
    <section>
      S
    </section>
    respectively.
  </body>
</article>

```

Figure 8: Sample Wikipedia semi-structured XML document

3.2 Query Collection

In previous years, different topic types have been used for the two main Adhoc

retrieval tasks at INEX [i.e. a distinction was made between Content Only (CO) and Content And Structure (CAS) topics]. These topic types have now been merged into one type: Content Only + Structure (CO+S) Topics. All information needed by the different Adhoc tasks and tracks are now expressed in the individual topic parts so only one topic type is needed. The CO+S topics consist of the following parts, which are explained in detail below:

<title> in which Content Only (CO) queries are given

<castitle> in which Content And Structure (CAS) queries are given

<description> a one or two sentence definition of the information need

<narrative> in which definitive definition of relevance and irrelevance are given

There are 107 CO+S queries in INEX 2007 V4 topic pool and 285 CO+S queries in INEX 2008 V4 topic pool. Figure 9 represents a typical CO+S query.

```
<!-- Topic definition -->
<inex_topic topic_id="414" ct_no="3">
<title>hip hop beat</title>
<castitle>//*[about(., hip hop beat)]</castitle>
<description>what is a hip hop beat?</description>
<narrative>To solve an argument with a friend about hip hop music and beats, I want to learn all there is to know about hip hop beats. I want to know what is meant by hip hop beats, what is considered a hip hop beat, what distinguishes a hip hop beat from other beats, when it was introduced and by whom. I consider elements relevant if they specifically mention beats or rythm. Any element mentioning hip hop music or style but doesn't discuss abything about beats or rythm is considered not relevant. Also, elements discussing beats and rythm, but not hip hop music in particular, are considered not relevant.
</narrative>
</inex_topic>
```

Figure 9: Sample topic from INEX 2007 v4 topic pool [Topic id = 414]

3.3 Retrieval Tasks and Evaluation Metrics

This section discussed the retrieval tasks for 2007 and 2008. The evaluation metrics for 2007 are discussed below. 2008 metrics are not yet finalized by INEX. The information in this chapter comes from [6] and [14].

3.3.1 Retrieval Tasks for 2007 and 2008

The University of Minnesota Duluth participates in the INEX Adhoc retrieval track. INEX 2007 focuses on retrieving variable-sized document parts, which represent XML passages or elements. Thus the result of retrieval can be either a passage or an XML element.

Within XML retrieval, results are regarded as relevant when they:

- contain relevant information (the result exhaustively discusses the topic), but
- contain as little non-relevant information as possible (the result is specific for the topic).

Following are the 2007 Adhoc retrieval tasks:

- ***Focused task:***

This task requires the system to return a ranked list of the most *Focused* elements or document parts. Also, this list cannot have any overlapping elements. This means that it cannot return both a paragraph and its container section. In this case, the system is forced to choose the most appropriate non-overlapping element that best represents the retrieval result.

- ***In-context task:***

This task assumes that the document is the most natural unit for retrieval. INEX 2007 distinguishes between two *in-context* tasks, depending on whether several document parts are returned or the result is a single answer per document.

2.1) *Relevant-in-Context:*

The purpose of this task is to return the most focused documents with the relevant parts highlighted. A constraint for this task is that the results should be non-overlapping.

2.2) *Best-in-Context:*

The purpose of this task is to return a single document part per document. The single document part represents the best entry point (BEP) for starting to read the relevant text in the document.

3.3.2 Relevance Assessments

INEX uses a highlighting assessment procedure to gather relevance assessments for the INEX topics. In this procedure, human assessors evaluate documents from the Wikipedia document collection and highlight sentences representing the relevant information. The INEX 2007 relevance assessment task involves recording the size of the highlighted text within the document part as well as the total text size of the document part in terms of number of characters. The relevance score of documents is calculated using these two parameters.

3.3.3 BEP Metric

In this section the metric used to evaluate the *Best-in-context* task is discussed. In

case of the *Best-in-context* task, the systems are expected to return a single document part representing the BEP for each relevant document. All of this information comes from [14].

The document score $S(d)$ for *Best-in-context* task is calculated with the distance similarity measure $s(x, b)$. $s(x, b)$ is a measure of how close the system-proposed entry point x is to the BEP b . Its value is 1 if the system-proposed value is the same as the BEP and it has a minimum value of zero. The controlling parameter A (for the distance between the actual BEP and the reported BEP) can be turned up to allow longer distances without much penalty, or down to reward systems which get very close to the actual BEP. $s(x, b)$ can be defined as follows:

$$s(x, b) = \frac{A \cdot L}{A \cdot L + d(x, b)}$$

Here,

A = controlling parameter for distance and $A > 0$

L = actual document length in characters.

$d(x, b)$ = distance between system-proposed value and BEP

INEX uses the $s(x, b)$ distance similarity score as an appropriate document score for the *Best-in-context* task:

$$S(d) = s(x, b)$$

The resulting $S(d)$ score varies between 0 (document without relevant text) and 1 (system-proposed value same as BEP).

According to [14], for the *Best-in-context* task, let,

D = a ranked list of documents

$S(d_r)$ = document score per document which lies between 0 and 1, inclusively

r = rank

N_{rel} = total number of relevant documents

$rel(d_r)$ = has value 1 if d_r contains highlighted text, 0 otherwise

Over the ranked list of documents, the following values are calculated:

1) Generalized precision $gP[r]$:

$$gP[r] = \frac{\sum_{i=1}^r S(d_i)}{r}$$

It is the sum of document scores up to a document-rank r , divided by the rank r , where 1

$\leq r \leq D$.

2) Generalized recall $gR[r]$:

$$gR[r] = \frac{\sum_{i=1}^r rel(d_i)}{N_{rel}}$$

It is the number of relevant documents retrieved up to a document-rank r .

These generalized measures are compatible with the standard precision and recall measures used in traditional information retrieval.

3) The average generalized precision for an INEX 2007 topic can be calculated by averaging the generalized precisions at natural recall points where generalized recall increases. (The generalized precision of non-retrieved relevant documents is zero.)

$$AgP = \frac{\sum_{r=1}^{|\mathcal{D}|} rel(d_r) \cdot gP[r]}{\sum_{r=1}^{|\mathcal{D}|} rel(d_r)} \cdot gR[|\mathcal{D}|]$$

4) Mean Average generalized Precision (MAgP) :

MAgP is simply the mean of the average generalized precision scores per topic.

Evaluation metrics for the 2008 collection are still under consideration and have not yet been finalized.

4. Parsing and Relevance Assessment Analysis

In 2007, the validation software from INEX, Checkrun, could not be installed due to a few erroneous documents and lack of essential information. Consequentially, the results submitted could not be validated before submission. These results contained certain invalid XPath. Since only valid XPath were accepted by the INEX online evaluation software, this had an adverse effect on our scores for the year.

4.1 Parsing the Wikipedia Collection

The parsing strategy of earlier years focused on a few tags, which were identified logically as the most meaningful tags (no statistics from relevance assessments were available). Table 2 gives the list of tags chosen as meaningful terminal nodes from 2007 and 2008, which is based on an analysis of relevance assessments 2007 v4, done during this research.

Each parse is generated using a set of terminal nodes defined in a configuration file. This file contains tags in a specific input format. It has 'tags to keep' and 'tags to index'. These two are the inputs for the parsing script. They identify which tags are to be kept in the XPath of the element and which are to be indexed. All the tags in 'tags to keep' are the tags which appear in the XPath of a given element. Prior to 2007, very few tags were kept to obtain the XPath of an element. However, since the INEX submission requires the complete path of an element starting from the root, all possible tags which could be identified as missing are put in this field. This set has to have all the tags from the collection in order to get an accurate XPath.

A tag is indexed if and only if this tag is present in 'tags to index.' The tags which

are meaningful and important with respect to context, are indexed. This year the set of tags was rejuvenated based on analysis of the relevance assessments. Refer to Table 2 to for the tags to index. Not all tags are meaningful. Indexing them all costs time and space. Retrieving elements from this set is expensive. Hence, the index has been limited to a small subset.

4.1.1 Article Parse

The highest level of parse generated is the article parse; it is identical to the 2007 article parse. The `<body>` tag is the only element recognized. All other tags are deleted. The element node for the article parse is the `<body>` tag. The body element is at the top of the document (its XPath is `'/article[1]/body[1]'`). A sample element produced by the article parse is shown in Figure 10.

```
<article>
<fno>10002</fno>
<doi>/article[1]/body[1]</doi>
<body>
  Emil Kraepelin ( February 15 1856 - October 7 1926) was a German Psychiatrist
  who attempted to create a synthesis of the hundreds of mental disorders classified by the
  19th century, grouping diseases together based on classification of common patterns of
  symptoms, rather than by simple similarity of major symptoms in the manner of his
  predecessors. In fact, it was precisely because of the demonstrated inadequacy of such
  methods that Kraepelin developed his new diagnostic system.
```

Figure 10: Sample output of article parse.

The `<article>` tag indicates the beginning of a parsed element. The `<fno>` tag holds the article number (its identifier in the Wikipedia collection). The `<doi>` tag is a formatting tag containing the XPath of the element. The `<body>` tag is a container for the

text enclosed in that element. The parsed element must be in this format before it can be converted to vector form by Smart.

4.1.2 Section Parse

The next level of parsing is done at the section level. The semi-structured Wikipedia documents raised issues in the section parsing. For example, there are section elements within the leaf nodes. Our section parse is modified so as to not include sections within terminal nodes, by first expanding all tags in the terminal nodes and then parsing for section elements. This process guarantees that no sections occur within terminal nodes in the parse results. Generating correct XPath expressions has been a challenge. The initial section parse focused on `<article>`, `<body>` and `<section>` tags. All the remaining tags were removed, though the text was retained. The terminal node for the section parse is the `<section>` tag.

To insure correct XPath expressions, all identifiable tags should be included during parsing. The code for these parses is portable and whenever new tags are identified they can be easily added to the configuration file. This gives complete XPath expressions for each element. The final run can be validated for correctness using the scripts for Checkrun. A sample section parse is shown in Figure 11.

```
<article>
<fno>1000</fno>
<doi>/article[1]/body[1]/section[1]</doi>
<body>
  Major novels The Poirot books take readers through the whole of his life in England,
  from the first book ( The Mysterious Affair at Styles ), where he is a refugee staying at
  Styles, to the last Poirot book ( Curtain ), where he visits Styles once again.
</name>Hercule Poirot
```

Figure 11: Sample output of section parse.

4.1.3 Para (Terminal node) Parse

Para parse is the most complex parse. The semi-structured nature of the Wikipedia documents allows nested terminal tags. The set for 2007 is {<p>, <normallist>, , , <numberlist>, <definitionlist>, <figure>, <table>}. For 2008, the set is {<p>, <name>, <emph3>, <figure>}. As before, the remaining tags were removed while retaining the text. An approach similar to that used in section parsing is used to get the correct para parse. The syntax of a few other special cases (an example of which is shown in Figure 12) are needed to get the complete path to the terminal node. These were specially parsed using string manipulations in Perl since a regular expression could be generalized only for a few other link tags. Tags in this category are <collectionlink>, <weblink>, <wikipedialink>, <unknownlink>, <outsidelink>, <language link>, <redirectlink>, <image> and <template>.

Another modification to parsing is indexing of name tags. Table 5 shows that the name tag is one of the most common elements in the 2007 relevance assessments. This was true for 2006 as well.

```
<collectionlink xlink:type="simple" xlink:href="23332.xml">
Pennsylvania
</collectionlink>

<unknownlink src="Douglas Briggs">
Douglas Briggs
</unknownlink>
```

Figure 12: Syntax of special cases in para parsing

Adding this element as an index term was statistically important. It is a formatting tag in

the Smart input format as well. Figure 13 shows a sample element from para parse. The last line shows <name> tag. Here, it is a formatting tag.

```
<article>
<fno>10005</fno>
<doi>/article[1]/body[1]/p[5]</doi>
<body>
  For a discussion of other educational philosophies, see educational philosophies and
  reforms
  <name> Educational progressivism
```

Figure 13: Sample output of para parse.

4.1.4 Para+mt Parse

The para + mt parse provides the input to Flex. The untagged text in documents has to be managed so that the correct document trees are built. This parse tags any untagged text with the <mt> and </mt> tags (this can be considered preprocessing required before the documents can be parsed). After this preprocessing, the documents are parsed using the regular para parser. Figure 14 shows a sample <mt> element after the para+mt parse.

```
<article>
<fno>10002</fno>
<doi>/article[1]/body[1]/section[1]/mt[1]</doi>
<body>
  For an extensive bibliography of English translations of Kraepelin's works see:
  http://www.kraepelin.org/\_wsn/page3.html See
  <name> Emil Kraepelin
```

Figure 14: Sample output of para+mt parse.

4.2 Relevance Assessment (2007 v4) Analysis

The total number of finalized topics for 2007 is 107. After the relevance assessments were released, an analysis was performed; results are shown in Appendix C. This analysis was performed from a *Best-in-context* view; Table 5 shows the result. The number of BEP's is 6,491. Data from 2006 relevance assessments show that the most frequently occurring elements in 2006 retain their positions in this table. BEP analysis helped us devise new strategies.

An important observation notes that BEPs tend to occur near the beginning of the documents. Using the relevance assessment data results in a considerable increase in performance (see Chapter 5 for details). The `<p>` and `<name>` tags alone comprise approximately 50 percent of all BEPs. This is the rationale for incorporating the `<name>` tag as an element to be indexed.

Appendix C shows a detailed view of BEPs per query. The number of BEPs per topic ranges from 2 to 479. The average number of documents in the pool provided to the relevance assessors in 2007 was close to 650. From this pool, assessors were instructed to mark the BEP of each document. While marking the relevant portions of the XML document under consideration, the assessor cannot see the formatting of the documents, i.e., it looks like a regular web-page to the user. Using the online tool for relevance assessment, X-Rai [15], the assessor colors the relevant portion(s). The BEP is marked using a tool provided by X-Rai. This appears as a bulls-eye image that is placed before the word which is chosen as BEP. We observed that human errors occurred while marking relevant documents.

Tag-name	Percentage	Frequency
p	25.311	1643
name	25.250	1639
emph3	9.650	627
collectionlink	6.650	432
body	6.020	391
title	4.945	321
image	4.570	297
caption	4.128	268
item	3.320	216
section	1.580	103
emph5	1.550	101
unknownlink	1.401	91
figure	1.010	66
outsidelink	0.908	59
articles	0.862	56
template	0.860	56
emph2	0.847	55
normallist	0.600	40
cell	0.100	11
indentation	0.046	3
td	0.00030	2
cadre	0.00030	2
blockquote	0.00030	2
conversionwarning	0.00030	2
row	0.00015	1
br	0.00015	1
value	0.00015	1
wikipedialink	0.00015	1
term	0.00015	1
div	0.00015	1
th	0.00015	1
gallery	0.00015	1
Total	100.00	6491

Table 5: Relevance assessment 2007 V4 analysis for Best Entry Point (BEP) elements

Table 6 provides a summary of an analysis performed on the 2007 relevance assessments. The tags *<name>*, *<p>*, *<section>* and *<emph3>* alone contribute approximately 81 % of the BEPs. The average number of BEPs per query is 60, with the number varying from 2 to 479.

Total number of topics 2007 v4	107
Total number of BEPs	6,491
Number of name + para + section + emph3	5,277
Other tags constituting BEPs	1,214
Mean	60.66
Median	36
Minimum BEPs / topic	2
Maximum BEPs / topic	479

Table 6: Summary of BEP analysis from relevance assessments 2007 v4

In the following chapter, we provide an analysis of the experiments performed to find good BEPs. The focus is on identifying a correct tag set and choosing a BEP located near the beginning of the document.

5. *Best-In-Context* Experiments

Analysis of the 2007 relevance assessments gives insight into the XML tags that are important for the *Best-in-context* task. Based on this analysis, we identified two other aspects which may lead to the successful identification of BEPs, namely, correlation and physical location.

We use two sets of tags in these experiments: the new tag set {p, name, emph3, body, section, figure} and the new tag set revised {p, name, emph3, body, section, figure, image, title, caption}. The tag <collectionlink> is left out of the collection since it contributes 2GB to the total size of the index.

Relevant-in-context and Flex files are the two input files used for these experiments. The Flex file contains a list of all the positively correlating elements found in a particular article, whereas the *Relevant-in-context* file contains a subset of these elements consisting of non-overlapping elements only. (Overlapping elements are those which contain both the parent and child node XPaths. For example, 1000168/article[1]/body[1]/figure[1] and 1000168/article[1]/body[1]/figure[1]/image[1] are said to be overlapping since 'image[1]' is a child node of 'figure[1]'. In a file with non-overlapping elements, only one of the two paths can exist.)

The slope and pivot used for paragraph retrieval are 0.12 and 15 for the new tag set and 0.12 and 14 for the new tag set revised, respectively. The slope and pivot for article retrieval are 0.04 and 120. The results are evaluated in term of MAgP against the 2007 v4 relevance assessments.

5.1 BEP based on Correlation Score

These experiments focus on the selection of BEP based on the correlation score.

Experiment 1: BEP as Highest Correlation Element and no restriction on the type of BEP

In this strategy, article retrieval is performed. These articles are ranked according to the correlation score. Flex is used to generate corresponding elements. The element with the highest correlation to the query is identified as the BEP for the given article. Physical location plays no role in the selection of the element. Slope and pivot for element retrieval are dependent on the tag set specified. Table 7 shows the best value (0.0439) is obtained when the number of articles is 400. Table 8 shows that the best value (0.0823) for the new revised tag set is obtained when the number of articles is 400 and the number of elements is 2,000 - 4,000.

Elements	100	250	500	750	1000	1250	1500	2000	4000	8000
Articles										
25	0.0227	0.0227	0.0198	0.0227	0.0227	0.0227	0.0227	0.0227	0.0227	0.0227
50	0.0253	0.0253	0.0240	0.0253	0.0253	0.0253	0.0253	0.0253	0.0253	0.0253
100	0.0318	0.0318	0.0278	0.0318	0.0318	0.0318	0.0318	0.0318	0.0318	0.0318
150	0.0357	0.0357	0.0305	0.0357	0.0357	0.0357	0.0357	0.0357	0.0357	0.0357
200	0.0379	0.0379	0.0328	0.0379	0.0379	0.0379	0.0379	0.0379	0.0379	0.0379
250	0.0403	0.0403	0.0374	0.0403	0.0403	0.0403	0.0403	0.0403	0.0403	0.0403
400	0.0439	0.0439	0.0430	0.0439	0.0439	0.0439	0.0439	0.0439	0.0439	0.0439
500	0.0420	0.0420	0.0420	0.0420	0.0420	0.0420	0.0420	0.0420	0.0420	0.0420

Table 7: Highest correlating element (no restriction on type of BEP);

Input: RIC; Index: New tag set

Table 9 shows that the best value (0.0995) is obtained for 400 articles and 2,000 – 4,000 elements. This experiment is identical to the experiment shown in Table 7; the only

Elements	100	250	500	750	1000	1250	1500	2000	4000	8000
Articles										
25	0.0612	0.0785	0.0803	0.0828	0.0617	0.0617	0.0617	0.0617	0.0617	0.0617
50	0.0648	0.0791	0.0798	0.0839	0.0673	0.0673	0.0673	0.0673	0.0673	0.0673
100	0.0706	0.0802	0.0825	0.0848	0.0736	0.0736	0.0736	0.0736	0.0736	0.0736
150	0.0731	0.0846	0.0872	0.0888	0.0762	0.0762	0.0762	0.0762	0.0762	0.0762
200	0.0756	0.0881	0.0912	0.0926	0.0788	0.0788	0.0788	0.0788	0.0788	0.0788
250	0.0785	0.0923	0.0947	0.0969	0.0809	0.0813	0.0817	0.0817	0.0817	0.0817
400	0.0797	0.0999	0.1008	0.1012	0.0817	0.0821	0.0822	0.0823	0.0823	0.0822
500	0.0792	0.0996	0.1099	0.1006	0.0812	0.0818	0.0819	0.0819	0.0819	0.0819

Table 8: Highest correlating element (no restriction on type of BEP);

Input: RIC; Index: New tag set revised

difference is that the input file used is the Flex file. Results are substantially better for Flex input. Table 10 shows that the best value (0.1024) is obtained at 400 articles and 2,000 – 4,000 elements. This experiment is identical to the experiment shown in Table 8, except that the input file is the Flex file. Best results for these four experiments result from use of the new tag set revised and Flex input.

Elements	100	250	500	750	1000	1250	1500	2000	4000	8000
Articles										
25	0.0749	0.0765	0.0783	0.0793	0.0801	0.0801	0.0801	0.0801	0.0799	0.0801
50	0.0762	0.0827	0.0834	0.0836	0.0839	0.0839	0.0844	0.0839	0.0849	0.0839
100	0.0796	0.0832	0.0836	0.0318	0.0876	0.0876	0.0876	0.0876	0.0876	0.0876
150	0.0821	0.0841	0.0848	0.0849	0.0884	0.0884	0.0884	0.0884	0.0884	0.0884
200	0.0836	0.0852	0.0864	0.0872	0.0901	0.0901	0.0901	0.0901	0.0901	0.0902
250	0.0850	0.0867	0.0880	0.0893	0.0912	0.0943	0.0956	0.0956	0.0956	0.0956
400	0.0967	0.0972	0.0973	0.0978	0.0983	0.0971	0.0993	0.0995	0.0995	0.0994
500	0.0964	0.0969	0.0972	0.0974	0.0977	0.0980	0.0984	0.0986	0.0985	0.0985

Table 9: Highest correlating element (no restriction on type of BEP);

Input: Flex; Index: New tag set

Elements	100	250	500	750	1000	1250	1500	2000	4000	8000
Articles										
25	0.0778	0.0785	0.0803	0.0828	0.0845	0.0845	0.0845	0.0845	0.0845	0.0845
50	0.0789	0.0791	0.0798	0.0839	0.0854	0.0854	0.0854	0.0854	0.0854	0.0854
100	0.0794	0.0802	0.0825	0.0848	0.0867	0.0867	0.0867	0.0867	0.0867	0.0867
150	0.0832	0.0846	0.0872	0.0888	0.0902	0.0902	0.0902	0.0902	0.0902	0.0902
200	0.0875	0.0881	0.0912	0.0926	0.0936	0.0936	0.0936	0.0936	0.0936	0.0936
250	0.0909	0.0923	0.0947	0.0969	0.0999	0.1007	0.1013	0.1013	0.1008	0.1007
400	0.0997	0.0999	0.1008	0.1012	0.1015	0.1019	0.1021	0.1024	0.1024	0.1023
500	0.0992	0.0996	0.1099	0.1006	0.1008	0.1012	0.1015	0.1013	0.1012	0.1013

**Table 10: Highest correlating element (no restriction on type of BEP);
Input: Flex; Index: New tag set revised**

Experiment 2: BEP as Highest Correlating Element and Element Set Membership

In this strategy, article retrieval is performed. The articles are ranked according to their correlation score. Flex is used to generate the corresponding elements. The element with highest correlation to the query and which belongs to the new tag set or the new tag set revised is identified as the BEP for the article under question.

In Experiment 1, we observe that the elements which highly correlate to the query are not necessarily classified as BEP. Using Table 5, tags in the BEP category are identified and ordered by frequency. Including all such tags is prohibitive. Hence, a few top-ranking tags (as per the BEP relevance assessments) were chosen to form what we call “new tag set.” The set was then revised to include a few more tags, creating the “new tag set revised.” The slope values used are discussed in the beginning of this chapter. Four experiments were performed. Only the input file and tag set change in these experiments.

Table 11 shows that the best value (0.0998) is obtained at 400 articles and 2,000 elements. Table 12 shows that the best value (0.1004) is obtained at 400 articles and 2,000-4,000 elements. The new tag set revised results are better.

Elements	100	250	500	750	1000	1250	1500	2000	4000	8000
Articles										
25	0.0587	0.0615	0.0597	0.0596	0.0596	0.0596	0.0596	0.0596	0.0596	0.0596
50	0.0684	0.0726	0.0726	0.0702	0.0701	0.0701	0.0701	0.0701	0.0701	0.0701
100	0.0744	0.0792	0.0809	0.0840	0.0813	0.0812	0.0812	0.0812	0.0812	0.0812
150	0.0768	0.0840	0.0863	0.0869	0.0894	0.0895	0.0895	0.0895	0.0895	0.0895
200	0.0780	0.0870	0.0890	0.0905	0.0913	0.0930	0.0932	0.0932	0.0932	0.0932
250	0.0774	0.0880	0.0899	0.0921	0.0929	0.0935	0.0955	0.0955	0.0955	0.0955
400	0.0774	0.0891	0.0929	0.0941	0.0956	0.0960	0.0966	0.0998	0.0968	0.0967
500	0.0774	0.0888	0.0925	0.0939	0.0953	0.0955	0.0960	0.0982	0.0960	0.0959

Table 11: Highest correlating element; Condition: BEP □ New tag set;

Input: RIC; Index: New tag set

Elements	100	250	500	750	1000	1250	1500	2000	4000	8000
Articles										
25	0.0795	0.0799	0.0805	0.0816	0.0820	0.0820	0.0820	0.0820	0.0820	0.0820
50	0.0813	0.0823	0.0826	0.0830	0.0831	0.0831	0.0831	0.0831	0.0831	0.0831
100	0.0835	0.0840	0.0846	0.0852	0.0859	0.0859	0.0859	0.0859	0.0859	0.0859
150	0.0884	0.0891	0.0898	0.9007	0.0905	0.0905	0.0905	0.0905	0.0905	0.0905
200	0.0945	0.0946	0.0951	0.0960	0.0965	0.0965	0.0965	0.0965	0.0965	0.0965
250	0.0968	0.0970	0.0977	0.0982	0.0994	0.0996	0.0999	0.0999	0.0999	0.0999
400	0.0976	0.0984	0.0988	0.0993	0.1001	0.1002	0.1003	0.1004	0.1004	0.1002
500	0.0974	0.0972	0.0979	0.0990	0.0998	0.1000	0.1002	0.1002	0.1002	0.1002

Table 12: Highest correlating element; Condition: BEP € New tag set revised;

Input: RIC; Index: New tag set revised

Table 13 shows that the best value (0.1078) is obtained at 400 articles and 2,000 elements. This experiment is identical to the experiment in Table 11 except that the input file is the Flex file. The new tag set is used here. Table 14 shows that the best value

(0.1256) is obtained at 400 articles and 2,000 elements. This experiment is identical to the experiment in Table 12 except that the input file is the Flex file. This experiment produces the best results to date, with Flex file input, new tag set, and choosing the BEP by selecting the highest correlating element that belongs to the new tag set.

Elements	100	250	500	750	1000	1250	1500	2000	4000	8000
Articles										
25	0.0831	0.0845	0.0849	0.0861	0.0874	0.0874	0.0874	0.0874	0.0874	0.0874
50	0.0857	0.0861	0.0872	0.0884	0.0892	0.0892	0.0892	0.0892	0.0892	0.0892
100	0.0889	0.0899	0.0908	0.0918	0.0925	0.0925	0.0925	0.0925	0.0925	0.0925
150	0.0923	0.0931	0.0939	0.0944	0.0949	0.0949	0.0949	0.0949	0.0949	0.0949
200	0.0956	0.0960	0.0967	0.0974	0.0981	0.0983	0.0983	0.0983	0.0983	0.0983
250	0.0971	0.0979	0.0977	0.1007	0.1012	0.1019	0.1028	0.1028	0.1028	0.1028
400	0.1033	0.1038	0.1043	0.1051	0.1067	0.1071	0.1072	0.1078	0.1077	0.1077
500	0.1027	0.1035	0.1038	0.1006	0.1047	0.1055	0.1064	0.1067	0.1067	0.1067

Table 13: Highest correlating element; Condition: BEP € New tag set;

Input: Flex; Index: New tag set

Elements	100	250	500	750	1000	1250	1500	2000	4000	8000
Articles										
25	0.0985	0.0994	0.1005	0.1027	0.1042	0.1042	0.1042	0.1042	0.1042	0.1042
50	0.0993	0.0998	0.0929	0.0943	0.1057	0.1057	0.1057	0.1057	0.1057	0.1057
100	0.1044	0.1053	0.1059	0.1074	0.1082	0.1082	0.1082	0.1082	0.1082	0.1082
150	0.1096	0.1096	0.1105	0.1108	0.1113	0.1113	0.1113	0.1113	0.1113	0.1113
200	0.1139	0.1141	0.1145	0.1150	0.1152	0.1152	0.1152	0.1152	0.1152	0.1152
250	0.1187	0.1193	0.1199	0.1206	0.1209	0.1216	0.1231	0.1231	0.1231	0.1231
400	0.1224	0.1231	0.1238	0.1242	0.1245	0.1246	0.1252	0.1256	0.1253	0.1253
500	0.1219	0.1208	0.1224	0.1229	0.1237	0.1241	0.1249	0.1251	0.1249	0.1248

Table 14: Highest correlating element; Condition: BEP € New tag set revised;

Input: Flex; Index: New tag set revised

5.2 XPath Collection

Based on the analysis of BEPs from 2007 assessments, it is observed that the BEP

usually lies towards the beginning of the article - a logical choice for both a reader and an assessor. Taking typical BEP location into account, we examine the feasibility of choosing the BEP based on this factor.

We begin by listing all the elements in a given document in the order of their physical occurrence. (The XPath of the first element in the document is listed first, etc.). A file listing of every XPath in each given document is created. This provides us with a way to identify elements by physical location.

Two new XPath collections were created for the INEX 2008 Wikipedia collection. The first lists elements from the new tag set and the second lists elements from the new tag set revised. These files were used for Experiments 3 and 4. Figure 15 shows a sample XPath file from the collection created.

```
1000168/article[1]/name[1]
1000168/article[1]/body[1]
1000168/article[1]/body[1]/figure[1]
1000168/article[1]/body[1]/figure[1]/image[1]
1000168/article[1]/body[1]/figure[1]/caption[1]
1000168/article[1]/body[1]/emph3[1]
1000168/article[1]/body[1]/p[1]
1000168/article[1]/body[1]/p[2]
1000168/article[1]/body[1]/p[3]
1000168/article[1]/body[1]/p[4]
1000168/article[1]/body[1]/p[5]
1000168/article[1]/body[1]/p[6]
1000168/article[1]/body[1]/p[7]
1000168/article[1]/body[1]/p[8]
1000168/article[1]/body[1]/p[9]
1000168/article[1]/body[1]/p[10]
1000168/article[1]/body[1]/p[11]
1000168/article[1]/body[1]/p[12]
1000168/article[1]/body[1]/p[13]
1000168/article[1]/body[1]/p[14]
1000168/article[1]/body[1]/p[15]
1000168/article[1]/body[1]/section[1]
1000168/article[1]/body[1]/section[1]/title[1]
```

Figure 15: Sample XPath file for document id 1000168.xml

Experiment 3: Selection of BEP based on Highest Physical Location

This experiment returns the element physically located near the beginning of the document as BEP. It is performed using both the *Relevant-in-context* and Flex files as input.

Table 15 shows the results based purely on physical position with the *Relevant-in-context* file as an input file. The best value (0.1405) is obtained at 400 articles and 8,000 elements. This results in a rank of 29 in INEX 2007 *Best-in-context* task. Table 16 shows that the best value (0.1729) for the new tag set revised is obtained at 400 articles and 4,000 elements. This result ranks 10th in the INEX 2007 *Best-in-context* task.

Elements	100	250	500	750	1000	1250	1500	2000	4000	8000
Articles										
25	0.0779	0.0852	0.0869	0.0869	0.0869	0.0869	0.0869	0.0869	0.0869	0.0869
50	0.0838	0.0954	0.0987	0.1013	0.1013	0.1013	0.1013	0.1013	0.1013	0.1013
100	0.0852	0.0964	0.0999	0.1011	0.1011	0.1011	0.1011	0.1011	0.1011	0.1011
150	0.0861	0.0968	0.1010	0.1022	0.1022	0.1022	0.1022	0.1022	0.1022	0.1022
200	0.0896	0.1081	0.1146	0.1196	0.1196	0.1196	0.1196	0.1196	0.1196	0.1196
250	0.0942	0.1095	0.1189	0.1221	0.1277	0.1292	0.1281	0.1281	0.1281	0.1281
400	0.0946	0.1118	0.1239	0.1250	0.1274	0.1303	0.1301	0.1363	0.1403	0.1405
500	0.0943	0.1115	0.1237	0.1247	0.1270	0.1299	0.1298	0.1352	0.1397	0.1399

Table 15: Highest physical position (no restriction on type of BEP);

Input: RIC; Index: New tag set

Table 17 shows the results using the Flex file as input (the element <article> was returned as BEP for every document). This experiment is identical to that in Table 15 except for the Flex input file. The poor results are due to the fact that the element <article> was returned in each case.

Elements	100	250	500	750	1000	1250	1500	2000	4000	8000
Articles										
25	0.1207	0.1228	0.1259	0.1283	0.1295	0.1295	0.1295	0.1295	0.1295	0.1295
50	0.1296	0.1334	0.1373	0.1388	0.1401	0.1401	0.1401	0.1401	0.1401	0.1401
100	0.1332	0.1346	0.1391	0.1398	0.1408	0.1408	0.1408	0.1408	0.1408	0.1408
150	0.1451	0.1485	0.1496	0.1536	0.1537	0.1537	0.1537	0.1537	0.1537	0.1537
200	0.1457	0.1502	0.1525	0.1578	0.1586	0.1586	0.1586	0.1586	0.1586	0.1586
250	0.1490	0.1532	0.1547	0.1583	0.1609	0.1612	0.1613	0.1621	0.1628	0.1629
400	0.1543	0.1588	0.1633	0.1682	0.1700	0.1704	0.1718	0.1720	0.1729	0.1724
500	0.1527	0.1575	0.1632	0.1693	0.1698	0.1705	0.1714	0.1721	0.1724	0.1723

**Table 16: Highest physical position (no restriction on type of BEP);
Input: RIC; Index: New tag set revised**

Elements	100	250	500	750	1000	1250	1500	2000	4000	8000
Articles										
25	0.0190	0.0190	0.0190	0.0190	0.0191	0.0192	0.0192	0.0192	0.0192	0.0192
50	0.0192	0.0192	0.0192	0.0192	0.0192	0.0193	0.0193	0.0193	0.0193	0.0193
100	0.0191	0.0191	0.0191	0.0191	0.0191	0.0192	0.0192	0.0192	0.0192	0.0192
150	0.0189	0.0189	0.0189	0.0189	0.0189	0.0189	0.0190	0.0190	0.0190	0.0190
200	0.0187	0.0187	0.0186	0.0186	0.0186	0.0186	0.0186	0.0187	0.0187	0.0187
250	0.0184	0.0184	0.0184	0.0184	0.0183	0.0184	0.0186	0.0184	0.0184	0.0184
400	0.0176	0.0176	0.0176	0.0176	0.0176	0.0175	0.0179	0.0176	0.0176	0.0176
500	0.0163	0.0163	0.0163	0.0163	0.0163	0.0163	0.0163	0.0163	0.0163	0.0163

**Table 17: Highest physical position (no restriction on type of BEP);
Input: Flex; Index: New tag set**

Experiment 4: Selection of BEP based on Highest Physical Location and Element Set Membership

This approach is identical to Experiment 3 except that an additional condition has been imposed for BEP selection, namely, that the BEP must belong to specified set (new tag set or new tag set revised).

Table 18 shows that the best value of 0.1389 is obtained at 400 - 500 articles and

4,000 elements. Table 19 shows that the best value (0.1722) is obtained at 400 articles and 4,000 - 8,000 elements; the BEP belongs to new tag set revised in this case. This result is virtually indistinguishable from the result obtained in Table 16.

Elements	100	250	500	750	1000	1250	1500	2000	4000	8000
Articles										
25	0.0853	0.0877	0.0926	0.0926	0.0926	0.0926	0.0926	0.0926	0.0926	0.0926
50	0.0744	0.0950	0.0985	0.1011	0.1148	0.1148	0.1148	0.1148	0.1148	0.1148
100	0.0752	0.0967	0.1078	0.1105	0.1169	0.1169	0.1169	0.1169	0.1169	0.1169
150	0.0776	0.0974	0.1015	0.1060	0.1180	0.1180	0.1180	0.1180	0.1180	0.1180
200	0.0788	0.0983	0.1061	0.1142	0.1206	0.1206	0.1206	0.1223	0.1223	0.1224
250	0.0790	0.1101	0.1108	0.1120	0.1242	0.1242	0.1242	0.1250	0.1250	0.1250
400	0.0833	0.1112	0.1238	0.1266	0.1320	0.1331	0.1331	0.1379	0.1389	0.1388
500	0.0838	0.1123	0.1239	0.1268	0.1339	0.1340	0.1340	0.1364	0.1389	0.1381

Table 18: Highest physical position; Condition: BEP € New tag set;

Input: RIC; Index: New tag set

Elements	100	250	500	750	1000	1250	1500	2000	4000	8000
Articles										
25	0.1203	0.1224	0.1256	0.1281	0.1292	0.1292	0.1292	0.1292	0.1292	0.1292
50	0.1293	0.1329	0.1369	0.1386	0.1399	0.1399	0.1399	0.1399	0.1399	0.1399
100	0.1328	0.1345	0.1387	0.1397	0.1403	0.1403	0.1403	0.1403	0.1403	0.1403
150	0.1448	0.1482	0.1492	0.1531	0.1533	0.1533	0.1533	0.1533	0.1533	0.1533
200	0.1453	0.1499	0.1520	0.1575	0.1584	0.1589	0.1589	0.1589	0.1589	0.1589
250	0.1487	0.1527	0.1544	0.1579	0.1608	0.1610	0.1611	0.1618	0.1625	0.1627
400	0.1537	0.1586	0.1628	0.1677	0.1697	0.1708	0.1713	0.1717	0.1722	0.1722
500	0.1524	0.1572	0.1630	0.1689	0.1693	0.1706	0.1716	0.1719	0.1721	0.1716

Table 19: Highest physical position; Condition: BEP € New tag set revised;

Input: RIC; Index: New tag set revised

5.3 Analysis

Experiments performed using the highest correlating element as BEP did not

perform well. The main reason for this is that the highest correlating element is not necessarily the best point to start reading the document. When analyzing our BEP result files based on highest correlation, we see that in most cases the element existed deep within the document. This behavior penalized us based on the evaluation metric (see Section 3.3.4) which computes the value based on the distance between the reported BEP and the actual BEP (the larger the distance, the lower the document score).

In general, the Flex file is not a good source of input for BEP experiments. Results of Experiments 3 and 4 clearly show that physical position is a dominating factor in identification of BEP. Overlapping elements in the Flex file guarantee that <article>, <name> and <body> (which occur at the beginning of the document) take precedence over other elements.

In addition to physical position, the tag set plays a major role in the recognition of the BEPs. This is clearly shown in Experiments 3 and 4.

6. Conclusions and Future Work

This chapter draws conclusions based on the analysis of the relevance assessments and the *Best-in-context* experiments. It also discusses the scope of future work in Section 6.2.

6.1 Conclusions

The analysis of the 2007 relevance assessments has greatly helped us improve performance. A major improvement is our ability to identify the elements (i.e., tags) which are important from a user's point of view. This is particularly important in identifying good BEPs, but the 2007 Adhoc tasks have also shown improvement with the new set of tags (see [2, 13] for the *Focused* and *Relevant-in-context* results).

Having analyzed the 2007 assessments enables us to identify the tags important from a *Best-in-context* task perspective. This analysis also shows that the BEP resides near the beginning of the document. This gave rise to a simple strategy of selecting the element with the highest physical position as BEP (see Experiment 3, Chapter 5). When comparing the results [0.0439 (highest correlation score, Table 7) to 0.1405 (for highest physical position, Table 15)], it is very clear that position alone produces a significant improvement. In experiments performed with the new tag set revised, the results improved further to 0.1729 (for highest physical position, Table 16). Thus, after having identified the target area for BEP, simply picking an element at the highest physical position can result in a better performance. The key is being able to identify the tags to be indexed.

Our results also show that the element with the highest correlation score is not

necessarily the BEP. None of the eight experiments performed using the highest correlating element as BEP show any significant results. Since the highest correlating element usually exists deep within the document, the distance between the actual BEP and the reported BEP increases, and as a result the document score decreases.

6.2 Future Work

The successful experiments performed were largely based on an analysis of the relevance assessments. All three subtasks for the Adhoc task show considerable improvement after the tags are revamped. Future work should include a more detailed analysis of the 2008 relevance assessments. Analysis of the BEP position and tag type will dominate the design of future experiments in this field.

Fine tuning of slope and pivot values with possible new tag sets may yield some improvements in BEP results. Close examination of the metric, especially with respect to penalty, may be useful as well.

For 2008, there are 285 topics versus 107 in 2007. The relevance assessments will present interesting challenges; analyzing them will be an uphill task in itself. However, this provides a way to better understand user behavior. This analysis will provide valuable insight into improving performance for all the Adhoc tasks.

References

- [1] Bakshi, V. Flexible Retrieval for the Semi-Structured Documents, MS Thesis, Department of Computer Science, University of Minnesota Duluth, 2006.
- [2] Bapat, S. Improving Results for Focused and Relevance-in-context tasks, MS Thesis, Department of Computer Science, University of Minnesota Duluth, 2008.
- [3] EvalJ
<http://sourceforge.net/projects/evalj/>
- [4] Ganapathibhotla, M. Query Processing in a Flexible Retrieval Environment. MS Thesis, Department of Computer Science, University of Minnesota Duluth, 2006.
- [5] INEX Document Collection
<http://www.inex.otago.ac.nz/data/documentcollection.asp>
- [6] INEX Run Submission
<http://www.inex.otago.ac.nz/tracks/adhoc/runsubmission.asp?action=specification>
- [7] Initiative for the Evaluation of XML Retrieval (INEX)
<http://inex.is.informatik.uni-duisburg.de/>
- [8] Kamps, J; Koolen, M; Lalmas, M. Where to Start Reading a Textual XML Document?
<http://staff.science.uva.nl/~kamps/publications/2007/kamp:wher07.pdf>
- [9] Kamat, N. Impact of untagged Text in Dynamic Element Retrieval, MS Thesis, Department of Computer Science, University of Minnesota Duluth 2007.
- [10] Khanna, S. Design and Implementation of a Flexible Retrieval System, MS Thesis, Department of Computer Science, University of Minnesota Duluth, 2005
http://www.d.umn.edu/cs/thesis/sudip_khanna_ms.pdf
- [11] List of Participants at INEX
<http://inex.is.informatik.uni-duisburg.de/2007/ShowParticipants.html>
- [12] Mone, A. Dynamic Element Retrieval for Semi-Structured Documents, MS Thesis, Department of Computer Science, University of Minnesota Duluth 2007.
- [13] Paranjape, D. Improving Focused Results, MS Thesis, Department of Computer Science, University of Minnesota Duluth, 2008.
- [14] Pehcevski, J; Kamps J; Kazai, G; Lalmas, M; Ogilvie, P; Piwowarski, B; Robertson, S. INEX 2007 Evaluation Measures (Draft)

<http://inex.is.informatik.uni-duisburg.de/2007/inex07/pdf/inex07-measures.pdf>

[15] Project: X-Rai

<https://developer.berlios.de/projects/x-rai/>

[16] Salton, G., Wong A., and Yang C. A vector space model for information retrieval. *Journal of American Society for Information Science*, 18(11):613-620, 1975

[17] Singhal, A; Buckley, C; Mitra, M. Pivoted Document Length Normalization. In Proc. Of the 19th Annual International ACM SIGIR Conference, (Zurich, 1996), 21-19.

[18] The World Wide Web Consortium (W3C)

<http://www.w3.org/>

[19] W3C XML 10 Years

<http://www.w3.org/2008/xml10/>

Appendix A

Step 1: Parsing

Parsing is done to convert XML documents to the smart input format. It is done at three different levels of hierarchy. For parsing, we need the parsing script and a config file. The config file contains the tags to keep and the tags to index.

Scripts are taken from /smart/08_modified_scripts/parsing_scripts

The main document collection is in /smart/wiki_docs/2008_collection_parsed

1.1) Parsing articles:

The script used to parse articles is wiki_parser.pl and the config file used for article parsing is nconfig_art.txt.

Usage: nohup perl wiki_parser.pl <config_file> <path_to_document_collection>
<path_to_output> <path_to_output_log_file > &

So, to parse articles in the document collection, the command looks like:

```
nohup perl wiki_parser.pl nconfig_art.txt /smart/wiki_docs/2008_collection_parsed/  
/smart/2008_collection/parsing/articles /smart/log.txt &
```

1.2) Parsing sections:

The script used to parse sections is section_parser.pl and the config file used for section parsing is nconfig_section.txt.

Usage: nohup perl section_parser.pl <config_file> <path_to_document_collection_part-
wise> <path_to_output> &

So, to parse sections in the document collection, the command looks like:

```
nohup perl section_parser.pl nconfig_section.txt
```

```
/smart/wiki_docs/2008_collection_parsed/part-0 /smart/2008_collection/parsing/sections  
&
```

1.3) Parsing paras:

The script used to parse sections is para_parserNT.pl and the config file used for section parsing is nconfig_para_no_mt.txt/nconfig_para_mt.txt/.

Usage: nohup perl para_parserNT.pl <config_file> <path_to_document_collection_part-wise> <path_to_output> &

So, to parse paras and mts in the document collection, the command looks like:

```
nohup perl para_parserNT.pl nconfig_para_mt.txt  
/smart/wiki_docs/2008_collection_parsed/part-0 /smart/2008_collection/parsing/paras &
```

Step 2: Indexing

Scripts required are:

1. generate_docloc.pl
2. generate_docid_docpath_mapping.pl
3. convert_rel_assessment_to_qrels_para_2005.pl.

Other files required are:

1. inex_query_loc
2. spec.parse_inex
3. make_inex_index_nnn
4. adhoc2007-assessments-v4-loc.txt

Folders to be created in the process of indexing are:

1. index_articles
2. index_sections
3. index_paras
4. all_element_index

Steps to be followed while indexing are:

2.1) Create all the above mentioned folders (the folders can have any name) and in each of the folder created, create a sub-folder and name it indexed (this name is important). After the folder 'indexed' is created in each of the four folders, copy make_inex_index_nnn into each of the indexed folder. Next, copy the file inex_query_loc and spec.parse_inex into the four folders (not in indexed, but one level higher in the hierarchy, for e.g. in index_articles folder).

2.2) Use the generate_docloc.pl script to create the inex_doc_loc file for index_articles, index_sections, index_paras. The inex_doc_loc file contains paths to each element. The scope of element here varies between articles, sections and paragraphs.

Usage: perl generate_docloc.pl <path_to_parsed_collection>
<path_to_inex_doc_loc_file>

Sample command:

```
perl generate_docloc.pl /smart/2008_collection/parsing/articles/  
/smart/2008_collection/indexing/articles/inex_doc_loc &
```

Similarly, `inex_doc_loc` is generated for section and para indices.

For All-element index, create one `inex_doc_loc` by merging the article `doc_loc`, section `doc_loc` and para `doc_loc` files.

After the completion of these steps, we will have a `inex_doc_loc` file in `all_element_index` that has paths to articles, sections and paragraphs.

Now, the `all_element_index` folder has `inex_doc_loc` file, `inex_query_loc` file, `spec.parse_inex` file and the `indexed` folder. The `indexed` folder has `make_inex_index_nnn` file.

2.3) Modify `make_inex_index_nnn` as follows:

- 1) Set `$current` to the path to the parent folder where the `indexed` folder belongs to
- 2) After Step 3.1, go to the end of the document. You can find a set of commented commands. For those commands, do the following:
 - 2.1) uncomment five lines under the line `#index` the collection. Make sure the four lines after the five lines are commented. Then run `nohup ./make_inex_index_nnn`
 - 2.2) next, comment the five lines that were uncommented above and uncomment the next two lines. Then run `nohup ./make_inex_index_nnn`.
- 3) The above two steps creates docs and queries. We need to create `qrels`.

3.1) there is a set of files that has to be converted into text format for the use by scripts. This process of conversion does not modify the original files but it just creates the necessary text files.

File conversion commands:

1. `/smart/smart.13.0/src/bin/smprint dict dict.words > dict.words.txt`
2. `/smart/smart.13.0/src/bin/smprint vec doc.nnn > doc.nnn.txt`
3. `/smart/smart.13.0/src/bin/smprint textloc textloc > textloc.txt`
4. `/smart/smart.13.0/src/bin/smprint vec query.nnn > query.nnn.txt`
5. `/smart/smart.13.0/src/bin/smprint inv inv.words > inv.words.txt`

3.2) Generate `docid_doc_path_mapping.pl`

Usage: `perl generate_docid_docpath_mapping.pl`

`/smart/2008_collection/indexing/articles/indexed/textloc.txt`

`/smart/2008_collection/indexing/articles/indexed/docid_docpath_mapping.txt &`

3.3) Next, we use the `convert_rel_assessment_to_qrels_para_2005.pl` script to create the `qrels.txt`. This script uses the `docid_docpath_mapping.txt`, `adhoc2007-assessments-v4-loc.txt`, and a dummy file to generate the output.

Usage: `perl convert_rel_assessment_to_qrels_para_2005.pl`

`<path_to_docid_docpath_mapping.txt> <path_to_rel-assessment-loc-file>`

`<dummy file> > <./qrels.txt>`

Sample command:

```
perl convert_rel_assessment_to_qrels_para_2005.pl  
/smart/2008_collection/indexing/articles/indexed/docid_docpath_mapping.txt  
/smart/2008_collection/indexing/articles/indexed/adhoc2007-assessments-v4-  
loc.txt /smart/2008_collection/indexing/articles/indexed/output.txt >  
/smart/2008_collection/indexing/articles/qrels.txt
```

Note: Make sure that the append operator > used so that the qrels.txt file is created.

3.4) Open the make_inex_index_nnn file and go to the end of file. Comment seven lines under #index the collection and uncomment the two lines after that.

nohup ./ make_inex_index_nnn. This creates the qrels files.

Step 3: Calculating Slope and Pivot:

3.1) To calculate Pivot:

File to use: pivot_unique_words.c

Things to change in this file:

- 1) Change NumDocs to the number of lines in the docid_docpath_mapping.txt file of the index for which you are doing the slope and pivot experiments.
- 2) Change the path to doc.nnn to the actual physical path of the doc.nnn of the index for which you are doing the experiments.

After these changes, compile this .c file. Command to compile the file is given in the file itself, and is as follows:

For GERARD machine the path of the above compile command line would be:

```
gcc -Wall -o pivot pivot_unique_words.c -I/smart/smart.13.0/src/h/ -  
L/smart/smart.13.0/src/lib -lretrieve -lfile -lgeneral -lproc -lprint -lindexing
```

This will generate the binary file pivot. Execute pivot to find the value of pivot for the collection. This will be the pivot value we will use.

3.2) To calculate Slope:

We will iterate over a range from 0.01 to 0.50, in increments of 0.01 to get the best slope value for the collection. To define best, we compare the output of our runs against the Trec slope and pivot (0.2 & 110 respectively) using the smprint utility provided in smart.

So this is what we do:-

Do the conversion and retrieval for the base case:

- 1) Open nnn_to_Lnu. This script does the conversion of nnn weights to Lnu. Change the paths in this script. Another thing to change in this file is the slope and pivot values. For base case, these are 0.2 and 110.
- 2) After running the above script (conversion script) for a slope-pivot value pair, run the retrieve_Lnu script. Before running it, change the paths in this script to your particular index.

Repeat this process for other cases. For other cases, the pivot will be the value you calculated above, and slope will go on from 0.01 to 0.50, in increments of 0.01.

Repeat this process for every slope-pivot value pair. After this is done, we will be ready to evaluate our results.

Step 4: Convert to INEX format

4.1) After getting the tr.Lnu.txt file (which is obtained from step 2.1.2 above), it should be converted to a .out file for Flex. The script that is used is smart_to_flex.pl at /smart/09_scripts/other_scripts/

Usage: nohup perl smart_to_flex.pl <path_to_tr.Lnu.txt> <path_to_.out file>

4.2) To convert it to focused, use gen_focused_output.pl at /smart/para0101/latest_scripts

Usage: nohup perl gen_focused_output.pl <path_to_the_.out_file>

<path_to_foc_output>

4.3) Convert to INEX format using convert_to_inex_v2007.pl

Usage: nohup perl convert_to_inex_v2007.pl <participantId> <unique_runId>

<task> <ResultType> <topN> <firstQueryId> <flexResFile> <outputFile>

Sample command:

nohup perl convert_to_inex_v2007.pl 72 Unique_ID BestInContext element 1500

414 <path_to_foc_output> <path_to_xml_file>

Step 5: Evaluate

5.1) Setting up EvalJ

1) Installed Perl with DB_File, Encode, XML::DOM, XML::Twig. Some of these could already be available.

2) Build a database of all elements with *global* offsets

Usage: perl inxdom2db.pl <databasefile> <corpusdir>

Here,

<databasefile> is output obtained after executing inxdom2db.pl

<corpusdir> the Wikipedia XML Corpus.

Sample command:

```
perl inxdom2db.pl DB_File /smart/wiki-docs/2008_collection/
```

Note: This assumes that the collection resides in subdirectories of <corpusdir>, e.g., like <corpusdir>/part-0/10002.xml.gz files may be zipped or unzipped. This will take a while, since it will parse the whole collection, and generate a database of 13 Gb...

Note: Use /smart/wiki-docs/2008_collection/. Do not use the parsed collection as DB creation will fail.

5.2) Evaluate

1) Evaluate Focused

Usage: perl foc_dom_eval2.pl [-q] [-c] <dbfile> <qrelmdir> <inxrun>+

with <dbfile> the database of (1), <qrelmdir> a directory containing the xml files with assessments, <inxrun> one or more runs.

Sample command:

```
nohup perl foc_dom_eval2.pl DB_File /smart/adhoc2007-assessments-v4/
/smart/07_with_06_parsing/indices/all-el/ret_0.12_18_1500/inxrun/focused-07-012-
18.xml
```

2) Relevant in Context

Usage: perl ric_dom_eval2.pl [-q] [-c] <dbfile> <qrelmdir> <inexrun>+ with <dbfile> the database of (1), <qrelmdir> a directory containing the xml files with assessments, <inexrun> one or more runs.

Sample command:

```
nohup perl ric_dom_eval2.pl DB_File /smart/adhoc2007-assessments-v4/  
/smart/07_with_06_pares/indices/all-el/ret_0.12_18_1500/inexrun/RIC-07-012-  
18.xml
```

3) Best in Context

Usage: perl bic_dom_eval2.pl [-q] [-c] <dbfile> <qrelmdir> <inexrun>+

With <dbfile> the database of (1), <qrelmdir> a directory containing the xml files with assessments, <inexrun> one or more runs.

Sample command:

```
nohup perl bic_dom_eval2.pl DB_File /smart/adhoc2007-assessments-v4/  
/smart/07_with_06_pares/indices/all-el/ret_0.12_18_1500/inexrun/BIC-07-012-  
18.xml
```

Appendix B

Best in Context Results Generation

B.1] Generate XPath Collection

Usage: perl create_xpath.pl <config_file> <path_to_document_collection>
<path_to_output>

B.2] *Best-in-Context* from *Relevant-in-Context*

The RIC files are at:

/smart/para0101Darshan/2008_FinalExperiments/07/rel_in_context/out

1) The BIC output is generated using the script gen_bic_from_ric.pl at
/smart/09_scripts/bic_scripts/

Usage: nohup perl gen_bic_from_ric.pl <path_to_ric_folder>
<path_to_xpath_coll>

Sample command:

```
nohup perl gen_bic_from_ric.pl  
/smart/para0101Darshan/2008_FinalExperiments/07/rel_in_context/out/100  
/smart/xpath_from_collection/2008_XPath/
```

Run the above script from the folder you want the bic.out to be created in.

2) Order the BIC output file we got in the previous step using the script `gen_bic-ric-mod.pl`.

Usage: `perl gen_bic-ric-mod.pl <path_to_ric_file> <path to the bic file>`

Sample command:

```
nohup perl gen_bic-ric-mod.pl
```

```
/smart/para0101Darshan/2008_FinalExperiments/07/rel_in_context/out/100/flex_1000  
.out /smart/mehta051/bic/bic_from_ric/bic_07/100/out/bic_flex_1000.out &
```

The output is generated in the same folder, ordered according to the prefix of its name.

After this step, follow Section 4.3 onwards from **Appendix A** for the complete evaluation.

B.3] *Best-in-Context* from Flex Output

The Flex files are at:

```
/smart/para0101Darshan/2008_FinalExperiments/07/flex_ot_nomt/out/
```

1) The BIC output is generated using the script `gen_bic_from_foc.pl` at

```
/smart/09_scripts/bic_scripts/
```

Usage: `nohup perl gen_bic_from_foc.pl <path_to_foc_folder>`

`<path_to_xpath_coll>`

Sample command:

```
nohup perl gen_bic_from_foc.pl
```

```
/smart/para0101Darshan/2008_FinalExperiments/07/rel_in_context/out/100
```

```
/smart/xpath_from_collection/2008_XPath/
```

Run the above script from the folder you want the bic.out to be present. After this step, follow Section 4.3 onwards from Appendix A for the complete evaluation.

Appendix C

Assessment	Total # of BEPs
414.xml	10
415.xml	58
416.xml	32
417.xml	19
418.xml	2
419.xml	20
420.xml	10
421.xml	13
422.xml	149
423.xml	87
424.xml	133
425.xml	58
426.xml	15
427.xml	61
428.xml	67
429.xml	97
430.xml	53
431.xml	57
433.xml	71
434.xml	51
435.xml	50
436.xml	14
439.xml	26
440.xml	76
441.xml	77
444.xml	13
445.xml	152

Table 20a

Assessment	Total # of BEPs
446.xml	24
447.xml	23
448.xml	125
449.xml	193
450.xml	62
453.xml	93
454.xml	66
458.xml	59
459.xml	38
461.xml	2
463.xml	15
464.xml	256
465.xml	65
467.xml	8
468.xml	5
469.xml	124
470.xml	14
471.xml	4
472.xml	12
473.xml	8
474.xml	35
475.xml	20
476.xml	45
477.xml	28
478.xml	27
479.xml	250
480.xml	80

Table 20b

Table 20a and Table 20b: Relevance assessment 2007 V4 analysis: BEPs per topic

Assessment	Total # of BEPs
481.xml	25
482.xml	92
483.xml	19
484.xml	10
485.xml	24
486.xml	10
487.xml	44
488.xml	114
489.xml	66
490.xml	19
491.xml	41
495.xml	17
496.xml	17
497.xml	28
498.xml	19
499.xml	479
500.xml	14
502.xml	63
503.xml	74
505.xml	37
506.xml	114
507.xml	72
508.xml	9
509.xml	267
511.xml	10
515.xml	62
516.xml	21

Table 20c

Assessment	Total # of BEPs
517.xml	21
518.xml	32
519.xml	79
520.xml	33
521.xml	14
522.xml	18
523.xml	36
525.xml	10
526.xml	19
527.xml	49
528.xml	289
529.xml	20
530.xml	90
531.xml	40
532.xml	139
533.xml	110
534.xml	28
535.xml	69
536.xml	34
537.xml	15
538.xml	100
539.xml	143
540.xml	143
541.xml	26
542.xml	10
543.xml	5

Table 20d

Table 20c and Table 20d: Relevance assessment 2007 V4 analysis: BEPs per topic (cont)