XML Element Retrieval@1CLICK-2

Atsushi Keyaki
Nara Institute of Science and Technology, Japan
atsushi-ke@is.naist.jp

Jun Miyazaki
Nara Institute of Science and Technology, Japan
miyazaki@is.naist.jp

Kenji Hatano
Doshisha University, Japan
khatano@mail.doshisha.ac.jp

Goshiro Yamamoto
Nara Institute of Science and Technology, Japan
goshiro@is.naist.jp

Takafumi Taketomi
Nara Institute of Science and Technology, Japan
takafumi-t@is.naist.jp

Hirokazu Kato
Nara Institute of Science and Technology, Japan
kato@is.naist.jp

ABSTRACT

In this paper, we try to apply an approach of XML element retrieval to 1CLICK-2. The aim of XML element retrieval is to identify key points in structured documents, and propose them to system users. We believe that the aim of XML element retrieval is largely the same as that of 1CLICK-2; enabling direct and immediate Information Access (DIIA). We report potentiality of usefulness of XML element retrieval to DIIA, and some difficulties to apply XML element retrieval to Web documents.

Team Name

NSTDB

Subtasks

1CLICK-2 (English)

Keywords

XML element retrieval, HTML document, re-structuration, document structure

1. INTRODUCTION

In this paper, we try to apply an approach of XML element retrieval to 1CLICK-2 [8] focused on direct and immediate Information Access (DIIA) [20] because the goals of DIIA and XML element retrieval are similar; both try to extract only relevant descriptions from documents beyond information retrieval.

Our research group participates the Initiative for the Evaluation of XML Retrieval (INEX) project[1], which is the largest community working on XML element retrieval. We mainly focus on the Ad hoc track for accurate search and try to exploit the experiences in 1CLICK-2.

In the past, availability of Web documents, or HTML documents, with XML element retrieval were not investigated in the INEX project though Wikipedia documents and scientific articles by IEEE computer society have been studied well. It is very important to survey whether the techniques of XML element retrieval are useful for HTML documents or not because HTML is a very popular format in structured documents. In other words, a scope of XML element retrieval is spread drastically if an effectiveness for HTML documents is proved, .

There are some challenges to apply the techniques of XML element retrieval to HTML documents. Firstly, strictness between HTML and XML differs very much in their grammars. HTML documents are lenient on a grammar, while XML requires a strict grammar. As a result, many of HTML documents cannot be parsed properly by an XML parser, which is an essential process for XML element retrieval. Therefore, we fix HTML documents and transform them into XHTML documents.

Secondly, most of HTML documents are relatively "shallow" in the depth of document structure whereas the average depth of the Wikipedia documents at INEX 2008 is 6.72 [7]. Techniques of XML element retrieval utilize document structures to judge which granular element is the most effective as a search result. An XML element retrieval system cannot extract appropriate elements if the possible granularity of elements are limited. Thus, we propose a re-struction method to deepen the document structures since some HTML tags potentially represent document structures and they can be extended to define nested structures.

Lastly, expected outputs between DIIA and XML element retrieval are different as an intimate matter. Intuitively, relevant descriptions for DIIA are data-centric contents, while those of XML element retrieval are document-centric contents. We should revise our strategy based on their goals.

The latter parts of this paper mainly focus on the first and second challenges. this is the first study adapting XML element retrieval techniques to HTML documents, as far as the authors' survey.

2. OVERVIEW OF XML ELEMENT RETRIEVAL

In this section, we describe the overview of XML elements and XML element retrieval techniques.

2.1 Comparison of Document Retrieval and Element Retrieval

Here, we explain the difference between XML element retrieval systems and well-used document retrieval systems. When many of document retrieval systems propose a list of relevant documents, the systems additionally provide users with result snippets [14], which are summaries of each document, approximately 50 words in length. Result snippets are generated by a text extraction technique that extracts the text that is nearby the query keywords. Search-system users

[1] https://inex.mmci.uni-saarland.de/

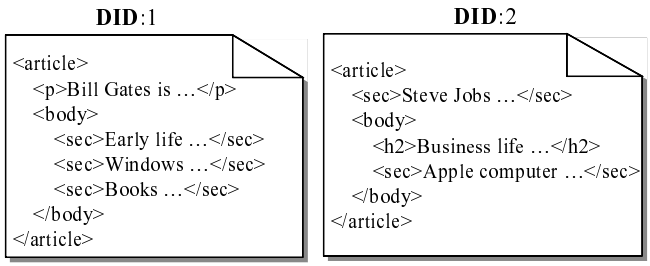

Figure 1: XML document

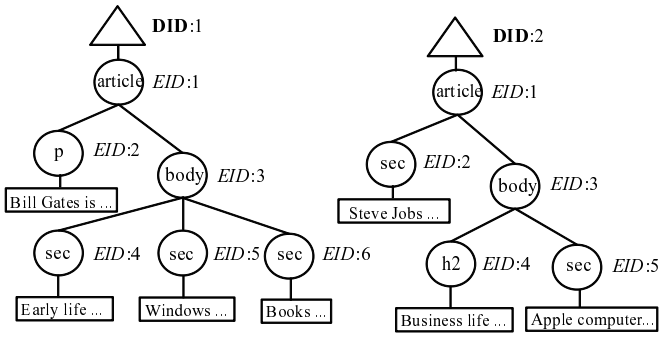

Figure 2: XML tree

utilize the result snippets located around the result when deciding which documents are worth browsing. Despite the fact that many search systems rely on result snippets, not all result snippets help them decide which documents to browse. This is because result snippets do not consider the context between the extracted text; as a consequence, some result snippets do not make sense [16].

On the other hand, the main purpose of an XML element retrieval is to extract the relevant descriptions (elements) from a query and propose them in descending order of their relevancy scores. XML element retrieval systems can propose a list that contains the relevant descriptions from a query, while many document retrieval systems propose a list that contains relevant documents for a query. Hence, users do not have to spend time seeking out the relevant parts that satisfy their information needs. This feature saves users' time and energy during information retrieval.

There are some other approaches to extract important descriptions in documents like automatic summarization, information extraction, passage retrieval and so on. Compared with these approches, one of the advantages of XML element retrieval is that search results are self-contained, which is one of the requirements for result snippets [14], because it considers the context of the sentences by utilizing document structures.

2.2 XML Element

We show concrete examples in Figures 1, 2, and 3 to explain the definition of XML elements. Figure 1 illustrates an example of XML documents. Each document is assigned a Document ID (DID). Figure 2 depicts trees that are translated from Figure 1. An XML document can be expressed

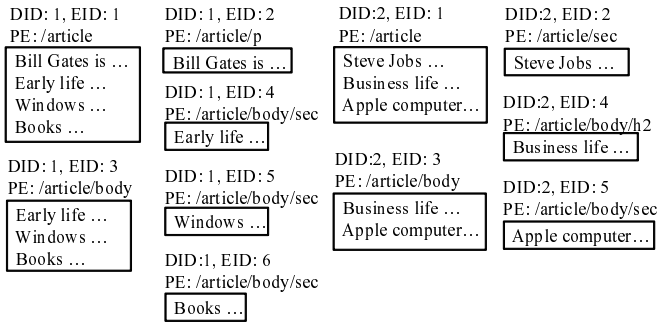

Figure 3: XML element

as a tree structure, which helps to understand the structure of the document. Each element is assigned an Element ID (EID), which is assigned in document order. We can identify an element by its DID and EID.

The author of a document makes structures, e.g., chapters, sections, paragraphs. We utilize these structures to identify the best description for satisfying users' information needs. In other word, we suppose that texts in an item are based on the same theme. Then, we try to extract the item that discusses the theme of users' interest.

A pair of start and end tags represents an XML element node in an XML tree, and the nested structure of XML elements represents ancestor-descendant relationships. Each element in Figure 3 is the text that is composed of a set of text nodes in the XML tree in Figure 2. This demonstrates why there are overlapping XML elements in XML documents.

We also describe the path expression (PE) of each element. Suppose a user seeks information about "Early life …", "Windows …" and "Books ⋯" of DID 1. XML element retrieval systems try to present an element whose EID is 3 in DID 1 to the user because the element contains all of the information that the user needs and no further information.

2.3 History of XML Information Retrieval

INEX project launched in 2002, and is the largest ongoing project for XML element retrieval. Test collections provided by the INEX project is widely used for evaluating the effectiveness of the XML element retrieval systems. The project also carries out a competition using XML documents generated by the scientific articles and the Wikipedia documents.

The INEX project requires search systems to return a non-overlapped ranked list as a search result in XML element searches. In addition, search systems extract 1,500 or fewer XML elements for each query. In the past, some existing studies do not remove overlapping elements and return a naïvely ranked list that is sorted in descending order of the XML elements' scores. We call such a list a simple ranked list. On the other hand, most studies have reported damage to search accuracy because of overlapping search results [9]: therefore, a ranked list without overlapping XML elements is also returned from recent XML element search systems. We call such a ranked list without overlaps a non-overlapped ranked list[2].

[2]XML element search systems can extract multiple XML

2.4 Targeted XML Documents

There are two types of XML documents: (1) data-centric which mainly contain single or compound term in their text nodes and (2) document-centric which tend to contain one or more sentences in their text nodes[1].

One of the most typical example of data-centric XML document is DBLP[3]. Studies investigating data-centric XML documents primarily focus on searching query keywords efficiently.

In contrast, the scientific articles and the Wikipedia documents are applicable as document-centric XML documents. With the documents, existing studies are performed for effective XML element retrieval. Needless to say, Ad hoc track of INEX project mainly targets document-centric XML documents.

In terms of XML type, HTML documents are classified as document-centric documents. It suggests that the framework of XML element retrieval may be applied to HTML documents. In this paper, we'd like to survey on it.

3. RELATED STUDIES

For effective XML element retrieval, some studies try to propose an accurate term-weighting scheme, while others try to remove unnecessary elements from search results.

3.1 Term-Weighting Schemes for XML Information Retrieval

The most important goal of XML element retrieval is highly accurate searches. The mainstream approach to extracting relevant elements is as follows: first, calculate a term weight for each element by using a term-weighting scheme; next, compute a score for each element using these term weights.

Term-weighting schemes for XML element retrieval are often derived from studies on document retrieval. Both of these are composed of three types of factors: local weights that are statistics derived from each document (element); global weights that are statistics derived from all document in a document set; and constant values (coefficients and parameters).

The most significant difference between document retrieval and XML element retrieval is the method for computing global weights. Term-weighting schemes in document retrieval assume that every document has the same attribute and belongs to the same class. Thus, global weights are calculated using all documents. However, in XML element retrieval, elements are assigned to classes. Global weights are calculated for elements of the same class. There are different ways to classify elements. One approach is to classify elements by path expression. In Figure 3, since Elements 4, 5, and 6 of Document 1, and Element 4 of Document 2 all have the same path expression `/article/body/sec`, the global weights are calculated using these elements.

Alternatively, elements with the same tag can be placed in the same class. Because Elements 4, 5, and 6 of Document 1 and Elements 2 and 4 of Document 2 all have the `sec` tag, the global weights are calculated using these elements, as depicted in Figure 3. We use classification based on path expression in our system, because this is reportedly more accurate [19].

There are several kinds of term-weighting schemes for XML element retrieval; e.g. TF-IPF [11], BM25E [12], and the query likelihood model for XML element retrieval [18] (QLMEL). BM25E is regarded as a more effective term-weighting scheme. Actually, most of the top-ranked search systems at INEX use BM25E [2]. We therefore utilize BM25E as a term-weighting schemes in this paper.

BM25E [12] is a probabilistic model. In a term calculation of the classic term-weighting scheme TF-IPF, statistics on the occurrence frequencies of terms are utilized. Conversely, BM25E utilizes not only the statistics but also element length (the number of terms in an element). The term weight $w(p, e, t)$ of term t in element e with path expression p is calculated as follows:

$$w(p,e,t) = \frac{(k_1+1)tf_{e,t}}{k_1((1-b)+b\frac{el_e}{avel_p})+tf_{e,t}} \cdot \log\frac{N_p - pf_{p,t} + 0.5}{pf_{p,t}+0.5} \quad (1)$$

where $tf_{e,t}$ is the term frequency of term t in element e, $pf_{p,t}$ is the element frequency of term t in the elements with p, N_p is the number of elements with p, el_e is the length of element e, and $avel_p$ is the average length of the elements with p. The parameters k_1 and b are set as 2.5 and 0.85 respectively based on the past experiments. Moreover, $s(e)$ is the score of e and is calculated as follows:

$$s(e) = \sum_{t_i \in T} w(p, e, t_i) \quad (2)$$

where T is a set of query keywords.

3.2 Data Cleansing Techniques

Document structures are useful to reveal the best description for users' information need. However, some structures are meaningless and elements defined by these structures are inappropriate as search results, and some existing studies try to eliminate them [6], [4].

In fact, removing useless or low-scored elements is effective for accurate XML retrieval. Although every granularity of XML elements should be treated as search targets, the effectiveness of the results decreases sharply if a search system returns non-informative XML elements. Extremely small XML elements are often not suitable for search results; Hatano et al. noted that when such meaningless XML elements are removed, the search accuracy improves [5]. Furthermore, our previous study suggests that large XML elements are also inappropriate for search results [10].

Those studies led to the fact that moderate granules are the most appropriate as search results, because extremely large granules (e.g. whole documents) tend to contain irrelevant descriptions and extremely small granules cannot satisfy the information need by themselves.

4. APPLYING XML ELEMENT RETRIEVAL TO HTML DOCUMENTS

Based on the discussion above, we work on adapting XML element retrieval techniques to HTML documents proveded by 1CLICK-2 organisers.

To achieve the end, we firstly re-structuration the HTML documents into well-formed XML documents to apply XML element retrieval techniques. Concretely, we complement

elements if these elements do not overlap with each other.

[3] `http://www.informatik.uni-trier.de/ ley/db/index.html`

tags to maintain the integrity of corresponding tags. Secondary, we re-structure the XML documents to deepen the document structures to utilize granularity of the documents maximally. Lastly, we also propose a method to eliminate unnecessary elements.

4.1 Pre-Processing for HTML Documents

Before the re-structuruction, we go through the following processes with the provided HTML documents.

1. removing attributes, comments, and special characters of HTML,
2. removing the stop words by SMART stop list[4], and
3. applying stemming step by Porter [15].

4.2 Validating Corresponding Relations in Tags

HTML documents need to be transformed into XML format because only a few original HTML documents can be parsed properly by an XML parser. To correct the corresponding relations in tags, we utilize CyberNeko HTML Parser[5]. The parser automatically complements the incomplete tags.

4.3 Re-Structuration of Document Structures

HTML tags, as well as XML tags, are categorized into two groups [21]. One represents structural classifications like `HTML`, `BODY`, `P` tags. The other indicates garnish of characters, specific contents, and meta-information like `B`, `TABLE`, and `A` tags, respectively. In terms of granularity, only structural tags are helpful.

Since XML tags are flexible, an automation classification method of the tags is required. Meanwhile, information of HTML tags is given and behavior of these tags is unique. Therefore, we manually classify the tags into the two groups. We list the structural tags as follows: `HTML`, `HEAD`, `TITLE`, `BODY`, `H1`, `H2`, `H3`, `H4`, `H5`, `H6`, `P`.

Heading tags (`H1`–`H6`) represent the granularity of contents, however, heading tags only contain not the contents but the topic name of the contents. We focus on the disagree between the potential granularity and document structures to deepen the document structures.

We explain the strategy of the re-struction with Figure 4. Basically, when a heading tag is discovered, a container heading tag (`CHX`[6], 1≤X≤6) is inserted before the heading tag. As a concrete example, Figure 4 depicts that the start tag for `CH2` is inserted before the start tag for `H2` colored red. From the second insertion of container heading tags, the behavior of the insertion bases on the magnitude relationship of heading tags' level. Supposed that the heading tag inserted earlier is `HX` and the heading tag inserted later is `HY` (1≤Y≤6).

There are three situations when the container heading tag is inserted as follows:

$X < Y$ When the level of the inserting heading tag is smaller than that of the inserted heading tag (the value Y is larger than X), a container heading tag is simply inserted just before the inserting heading tag. Concretely, in Figure 4, the level of the `H3` tag colored blue is smaller than that of the `H2`. Thus, the start tag for `CH3` is inserted before the start tag for `H3`.

$X = Y$ In case the values X and Y are equal, before inserting the container heading tag, the previous container heading tag is closed. Because the levels of the `H3` tag colored blue and the `H3` tag colored orange is the same, the end tag for `CH3` is inserted followed by the start tags for `CH3` and `H3` in Figure 4.

$X > Y$ If the level of the inserting handling tag is larger than that of the inserted heading tag, the inserted container heading tags are closed while the level of the inserting heading tag is the same or smaller compared with that of the inserted container heading tag. When the `H1` tag colored green is to be inserted in Figure 4, the `CH2` tag and `CH3` tag have been inserted. Since the level of `H1` is larger than that of these two tags, the end tags for `CH3` and `CH2` are inserted before the start tag for `CH1` is inserted.

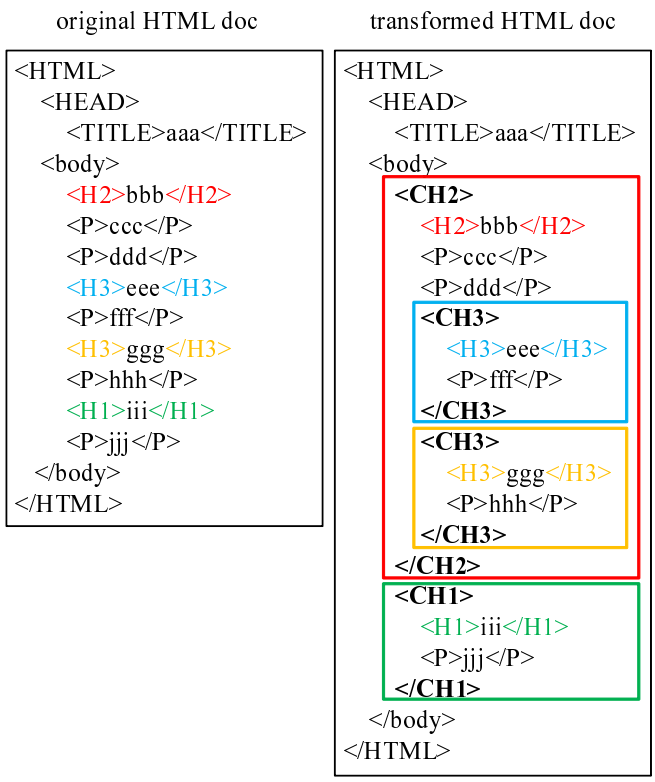

Figure 4: Re-structuring HTML documents

With the re-struction method, the depth of the documents can be deepened. To investigate the effectiveness of the method, we apply the method to provided HTML documents. Table 1 indicates the number of the max depth of original documents and documents. The deeper the max depth, the larger the number of the transformed documents. As a result, the average depth of the document set is increased from 3.02 to 5.18.

[4] `http://jmlr.csail.mit.edu/papers/volume5/lewis04a/a11-smart-stop-list/english.stop`
[5] `http://nekohtml.sourceforge.net/`
[6] The value of X is the same value as the heading tag.

Table 1: Depth of the document structure

max depth	original	transformed
1	217	217
2	102	102
3	25647	3110
4	890	4126
5	53	7461
6	7	8051
7	1	3359
8	6	462
9	3	38
average	3.02	5.18

Table 2: Accuracy rates of the main body check

	threshold τ					re-structuration	
naïve	0.1	0.3	0.5	0.7	0.9	naïve	τ (0.7)
.53	.80	.73	.80	.80	.73	.27	.60

Some other tags like `UL`, `OL`, `DL`, `TABLE`, `FORM`, `DIV` are also regarded as structural tags. Managing these tags is one of our future works though these tags can by noise when these tags are used as they are.

Furthermore, `BR` and `HR` separate contents, which may be a clue of re-struction. This is also a part of our future works.

4.4 Eliminating Noise Elements

As we remarked earlier, extremely small elements are not appropriate as search results. Structured documents including XML documents are composed of some components, i.e., main body, table-of-contents, references and so on. The other contents except the main body are basically consists not of sentences but of keywords. These descriptions do not tend to satisfy an information need directly, even though they can serve as navigational information. Therefore, we can remove outside of the main body for effective retrieval, of which idea is adaptable to HTML documents. In XML documents, outside of the main body are removed with the limitation of the element length of the number of distinct terms, whereas, HTML documents are difficult to eliminate all "noises" because HTML documents are dirtier and contain more complex noises. Consequently, we propose a method to remove the noises from HTML documents.

Suppose that outside of the main body contains many hyperlinks. Based on this hypothesis, we compute the ratio of the text size in the tag for A divided by the total text size of the elements. If the ratio exceeds the threshold value τ, the element is eliminated as a noise.

We performed a preliminary experiments to investigate the validity of the noise classification method. We manually judge elements in the top 15 for a query "michael jackson death" whether each element is the main body or not, with changing τ. Table 2 depicts the accuracy rates. As we can see from the table, the accuracy rates improved when we set the threshold τ compared with the naïve method. Note that the accuracy rates mean not precision but the correctness of the main body.

We also surveyed the effects on the re-structuration of the document structures discussed in Section 4.3. The re-structuration cause dropping the accuracy rate drastically, however, it can recover the accuracy rate with the noise classification method.

4.5 Generating Search Results

There are two purposes for this study. One is to take a challenge whether XML element retrieval techniques can be properly adapted to HTML documents or not. The other is to investigate whether XML element retrieval is applicable to DIIA task or not. Thus, we introduce the way of INEX to generate search results, or nuggets.

There are three types of tasks in the Ad hoc track at INEX 2008 [7] as follows:

- Focused Task,
- Relevant in Context Task, and
- Best in Context Task.

For the Focused Task, a ranked-elements-list of non-overlapping results must be returned. It is evaluated at early precision relative to the highlighted (or believed relevant) text retrieved.

For the Relevant in Context Task non-overlapping results (elements or passages) must be returned, these are grouped by document. It is evaluated by mean average generalized precision where the generalized score per article is based on the retrieved highlighted text.

For the Best in Context Task a single starting point (element's starting tag or passage offset) per article must be returned. It is also evaluated by mean average generalized precision but with the generalized score (per article) based on the distance to the assessor's best-entry point.

We submitted following six runs for 1CLICK-2.

EF Output is generated in the same manner as the Focused Task.

ER Output is generated grouped by document based on element score (the same manner as the Relevant in Context Task).

DR Output is generated grouped by document based on document score (the same manner as the Relevant in Context Task).

EB Only one element with the highest score is retrieved based on element score (the same manner as the Best in Context Task).

DB Only one element with the highest score is retrieved based on document score (the same manner as the Best in Context Task).

mobileEF A way of generating the result is the same as EF though text size can be retrieved is limited.

4.6 Outputs and Discussions

Unfortunately, the results of our formal runs at 1CLICK-2 are very poor. This results indicates that it is difficult to succeed in the DIIA task by simply adapted XML element retrieval approach. In next step, we consider an appropriate method to generate nuggets from elements with high score. Studies from other team at 1CLICK [13], [3], [17] may be helpful.

5. CONCLUSION

This is the first study applying XML element retrieval to Web documents, or HTML documents. We found some problems above it; HTML documents are too dirty in their grammars to parse properly, and their heights are relatively low to utilize granularity of XML elements fully. In addition, noises in HTML documents are more complex compared with well-formatted XML documents.

To solve these problems, we propose the re-struction method to deepen document structures and the noise classification method leverage hyperlink in HTML documents. Though the scale of the preliminary experiments are extremely small and limited, we found the potential power of these methods. We are going to operate exhaustive experiments with larger setting.

As a result of our formal runs at 1CLICK-2, it turns out that our approach, XML element retrieval, is not appropriate to generate nuggets from a ranked-element-list. This remains for one of future works.

As another future works above applying XML element retrieval to HTML documents, we consider another approach to re-stcuture HTML document and the evaluation method for our challenge.

6. ACKNOWLEDGMENT

This work was partly supported by Grant-in-Aid for JSPS Fellows and JSPS KAKENHI Grant #22240005, #23500121, and #22700248.

7. REFERENCES

[1] H. Blanken, T. Grabs, H.-J. Schek, R. Schenkel, and G. Weikum. *Intelligent Search on XML Data: Applications, Languages, Models, Implementations, and Benchmarks*, volume 2818 of *LNCS*. Springer-Verlag, 2003.

[2] S. Geva, J. Kamps, and A. Trotman. *Advances in Focused Retrieval*. Springer Berlin, 2009.

[3] T. S. H. T. Hajime Morita, Takuya Makino and M. Okumura. TTOKU Summarization Based Systems at NTCIR-9 1CLICK task. In *Proc. of the 9th NTCIR Conference*, 2011.

[4] K. Hatano, H. Kinutani, T. Amagasa, Y. Mori, M. Yoshikawa, and S. Uemura. Analyzing the Properties of XML Fragments Decomposed from the INEX Document Collection . In *Advances in XML Information Retrieval*, volume 3493 of *LNCS*, pages 168–182. Springer Berlin, 2005.

[5] K. Hatano, H. Kinutani, M. Watanabe, Y. Mori, M. Yoshikawa, and S. Uemura. Keyword-based XML Portion Retrieval: Experimental Evaluation based on INEX 2003 Relevance Assessments. In *Proc. of INEX 2003 Workshop*, 2004.

[6] F. Huang, S. Watt, D. Harper, and M. Clark. Compact Representations in XML Retrieval. In *Formal Proc. of INEX 2006 Workshop*, volume 5631 of *LNCS*, 2007.

[7] J. Kamps, S. Geva, A. Trotman, A. Woodley, and M. Koolen. Overview of the INEX 2008 Ad Hoc Track. In *Formal Proc. of INEX 2008 Workshop*, volume 5631 of *LNCS*, 2009.

[8] M. P. Kato, M. Ekstrand-Abueg, V. Pavlu, T. Sakai, T. Yamamoto, and M. Iwata. Overview of the NTCIR-10 1CLICK-2 Task. In *Proc. of the 10th NTCIR Conference*, 2013.

[9] G. Kazai, M. Lalmas, and A. P. de Vries. The Overlap Problem in Content-Oriented XML Retrieval Evaluation. In *Proc. of the 27th ACM SIGIR*, 2004.

[10] A. Keyaki, J. Miyazaki, and K. Hatano. A Method of Generating Answer XML Fragment from Ranked Results. In *INEX 2009 Workshop Pre-Proceedings*, 2009.

[11] F. Liu, C. Yu, W. Meng, and A. Chowdhury. Effective Keyword search in Relational Databases. In *Proc. of ACM SIGMOD*, 2006.

[12] W. Liu, S. Robertson, and A. Macfarlane. Field-Weighted XML Retrieval Based on BM25. In *Formal Proc. of INEX 2005 Workshop*, volume 3977 of *LNCS*, 2006.

[13] K. T. Y. S. T. Y. H. O. Makoto P. Kato, Meng Zhao and K. Tanaka. Information Extraction based Approach for the NTCIR-9 1CLICK Task . In *Proc. of the 9th NTCIR Conference*, 2011.

[14] C. D. Manning, P. Raghavan, and H. Schutze. *Introduction to Information Retrieval*, pages 157–159. Cambridge University Press, 2008.

[15] M.F.Porter. An Algorithm for Suffix Stripping. In *Computer Laboratory, Cambridge*, 1980.

[16] S. Nakamura. Trustworthiness Analysis of Web Search. *Journal of Japanese Society for Artificial Intelligence*, 23(6):767–774, 2008.

[17] Y.-I. S. Naoki Orii and T. Sakai. Microsoft Research Asia at the NTCIR-9 1CLICK Task. In *Proc. of the 9th NTCIR Conference*, 2011.

[18] P. Ogilvie and J. Callan. Parameter Estimation for a Simple Hierarchical Generative Model for XML Retrieval. In *Formal Proc. of INEX 2005 Workshop*, volume 3977 of *LNCS*, 2006.

[19] B. Piwowarski and P. Gallinari. A Bayesian Framework for XML Information Retrieval: Searching and Learning with the INEX Collection. *Journal of Information Retrieval*, 8(4):655–681, 2005.

[20] T. Sakai, M. P. Kato, and Y.-I. Song. Click the Search Button and Be Happy: Evaluating Direct and Immediate Information Access. In *Proc. of the 16th ACM CIKM*, 2011.

[21] T. Tokuda and K. Tajima. Classification of XML Tags according to Their Roles in Document Structure. *DBSJ Journal*, 8(1):1–6, 2009.