Hindawi Publishing Corporation
The Scientific World Journal
Volume 2014, Article ID 404518, 10 pages
http://dx.doi.org/10.1155/2014/404518

Research Article

An Exponentiation Method for XML Element Retrieval

Tanakorn Wichaiwong

Department of Computer Science, Faculty of Science, Kasetsart University, Bangkok 10903, Thailand

Correspondence should be addressed to Tanakorn Wichaiwong; tanakorn1977@gmail.com

Received 8 August 2013; Accepted 5 December 2013; Published 13 February 2014

Academic Editors: J. Shu and F. Yu

XML document is now widely used for modelling and storing structured documents. The structure is very rich and carries important information about contents and their relationships, for example, e-Commerce. XML data-centric collections require query terms allowing users to specify constraints on the document structure; mapping structure queries and assigning the weight are significant for the set of possibly relevant documents with respect to structural conditions. In this paper, we present an extension to the MEXIR search system that supports the combination of structural and content queries in the form of content-and-structure queries, which we call the Exponentiation function. It has been shown the structural information improve the effectiveness of the search system up to 52.60% over the baseline BM25 at MAP.

1. Introduction

Nowadays, the XML (http://www.w3.org/TR/xml11/) research is willing increasingly more documents having the structure with respect to certain structural [1]. Exploiting this structure is a significant part of improving retrieval effectiveness which can be divided into two categories: using document structure and user queries. Several form of the document's structure based retrieval models have been developed, such as BM25F [2] ranking function that is composed of several document fields with potentially different degrees of importance; PRM-S [3] is based on probabilistic retrieval model; and FRM [4] is the relevance feedback function based on the language model. Broschart and Schenkel presented the proximity weighting to improve the search system [5]. On the other hand, it is based on user queries, such as QRX [6] which is based on tree matching model without knowing the exact structure of the data, using the similarity measure of the vector space model. Unfortunately, this method has a drawback on the efficiency issue. The weight has been based on depth of the path and location in the document logical structure and then used as probabilities function based on the language model [7]; the length has been used as a normalization incorporated through a prior probability in the ranking function [8]. In [9, 10], highlight the structure weight in TopX (http://topx.sourceforge.net/) search engine. It assigns a small constant and tunable score for every navigational condition that is matched to query by using the frequency of the tag name. The weight has also been calculated based on the distribution of tag names which is used in a way similar to the binary independence retrieval model, but investigating the presence of tags in relevant and nonrelevant elements, to estimate the tag weights [11]. In [12], it is shown the structure does not improve the effectiveness of the retrieval system much because the users are very bad at giving structural hints with respect to INEX-IEEE collection and it requires further investigation. In this paper, we are investigating retrieval technique and related issues over a strongly structured collection of XML documents with the Initiative for the Evaluation of XML Retrieval (INEX) (https://inex.mmci.uni-saarland.de/) collections based on user queries. With richly structured XML data, we have been shown that the structural information using the Exponentiation function could be utilized to improve the effectiveness of search systems.

This paper is organized as follows. Section 2 reviews the data model and notions. Section 3 explains the presents state of the art approaches. Section 4 shows the experiment results and discussion; conclusions and further work are drawn in Section 5.

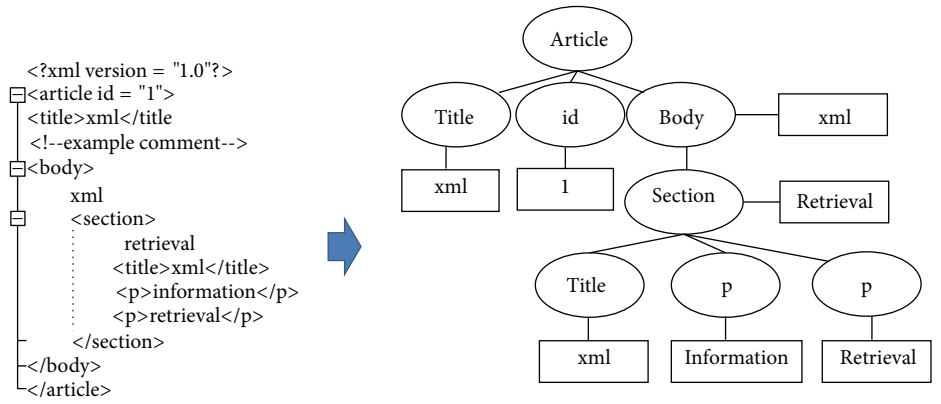

Figure 1: The Example of XML Element Tree.

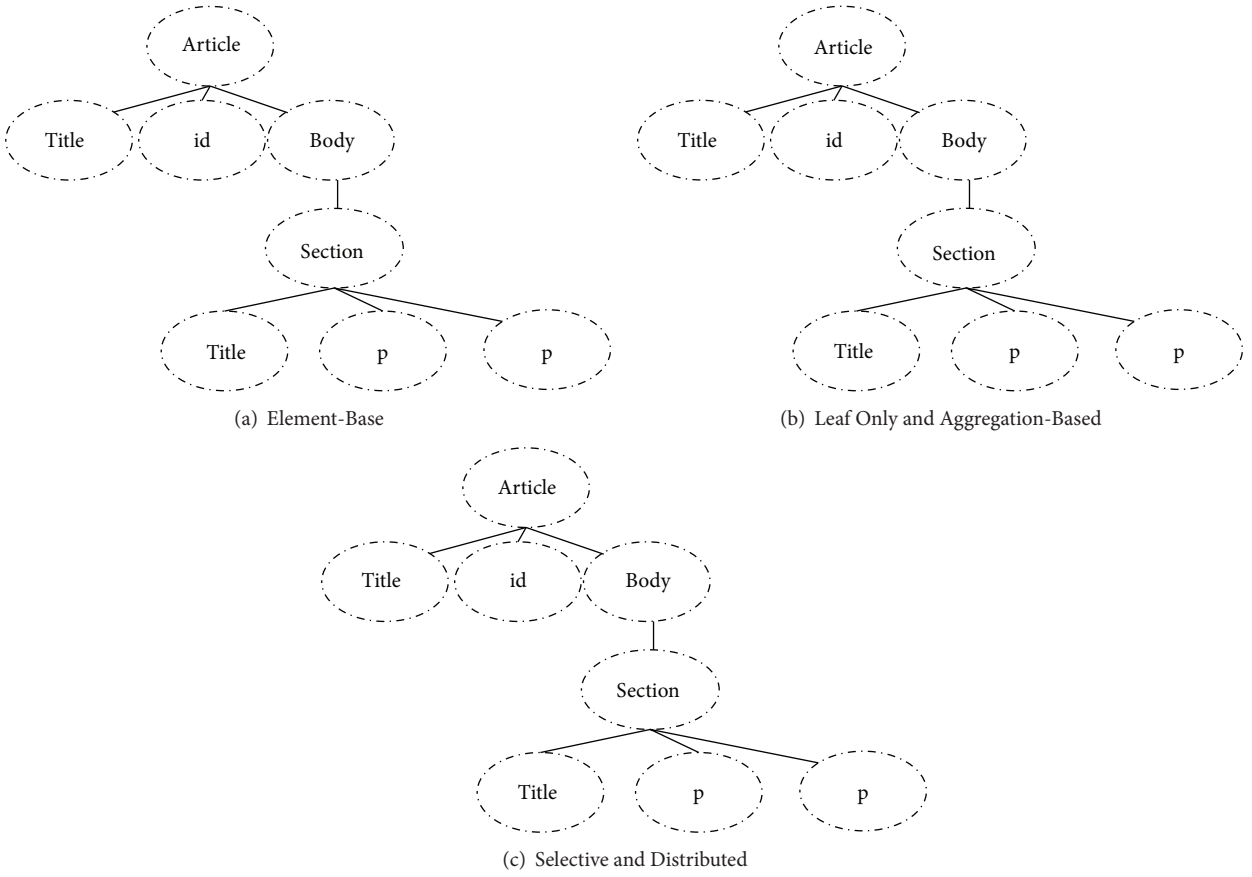

Figure 2: Illustrations of some of the indexing strategies.

2. Data Model and Notions

In this section, we provide some historical perspectives on areas of XML research that have influenced this article as follows.

2.1. XML Indexing Methods. The basic XML data model is a labeled, ordered tree. Figure 1 shows the data tree of an XML document based on the node-labeled model.

Classical retrieval models have been adapted to XML retrieval. Several indexing strategies have been developed in XML retrieval as shown in Figure 2.

Element Base indexing [8] allows each element to be indexed on the basis of both direct text and the text of descendants. This strategy has a major drawback in that it is highly redundant. Text occurring at the nth level of the XML logical structure is indexed n times and thus requires more index space. This strategy is illustrated in Figure 2(a), where all elements are indexed. Leaf-Only indexing [13] allows

indexing of only leaves through element or elements directly related to text. This strategy addresses the redundancy issues noted above. However, the propagation algorithm for the retrieval of nonleaf elements requires a certain level of efficiency. This strategy is illustrated in Figure 2(b), where the leaf elements are indexed. Aggregation-Based indexing [14] uses the concatenated text of an element to estimate a term statistic. This strategy has been used to aggregate term statistics directly on the basis of the text and its descendants. This is illustrated in Figure 2(b), where the leaf elements are indexed. Selective indexing [13, 15] involves eliminating small elements and elements of a selected type; this strategy is illustrated in Figure 2(c), where only semantic elements are indexed. Distributed indexing [15] is separately created for each type of element in conjunction with the selective indexing strategy, as shown in Figure 2(c). The ranking model runs each index separately and retrieves ranked lists of elements. These lists are merged to provide a single rank across all element types. To merge lists, normalization is performed to take into account the variation in elements size across the different indices such that scores across indices are comparable.

2.2. XML Query Languages. Querying in structured documents must be with respect to content and structure. INEX identified two types of queries [23, 24]; they are content only (CO) and content and structure (CAS) as follows.

2.2.1. Content Only Queries. These queries are formed by ignoring the document structure, in the same way as the traditional queries used in IR collections. However, they pose a challenge to XML retrieval in that the retrieval results in returning document components, that is, XML elements instead of whole documents in response to a user query. Queries can be elements of various complexities, that is, at different levels of the XML document's structure. This is suitable for XML retrieval where users do not know or are not concerned about the structure, that is, with the logical organization of the document, when expressing their information needs. For example, the best answer for a query "XML retrieval" applied to Figure 1 may be a "section" and not "title" or "p" elements.

2.2.2. Content-and-Structure Queries. These queries contain conditions of both content and structure. These conditions may refer to the content of specific elements and specify the type of requested answer elements. However, the complexity and the expressiveness of content-and-structure query languages are difficult for the end users because they have to know the logical organization of the document when expressing their information needs. Trotman and Lalmas [12] showed that the structure did not improve the effectiveness of the retrieval system very much because users were normally not capable of giving useful structural hints with respect to INEX-IEEE collection. However, the content-and-structure query can be very useful for expert users in specialized scenarios.

2.2.3. The Narrowed Extended XPath I. The Narrowed Extended XPath I (NEXI) query language was developed at INEX [25] as a simple query language for content-oriented XML retrieval evaluation. The enhancement comes from the introduction of a new function named "about()". The "contains()" function of XPath, which requires an element (its text) to contain the given string content, was replaced by the "about()" function, which requires an element to be about the content. The NEXI query provides support for the descendant axis as follows. $//T[t]$ is simple elements with paths matching T and contents about t. $//S[s]//T$ returns elements T which are descendants of the element S, where the element S contains s. $//S[s]//T[t]$ returns elements T which are descendants of the element S, where the element S contains s and the element T contains t.

2.3. Structure Weight IR. Schlieder and Meuss presented the QRX [6] which is based on tree matching without knowing the exact structure of the data of the similarity measure of the vector space model; an element score is computed as follows:

$$\text{Score}\,(e,q) = \sum_{t\in q} tf_t * idf_t. \tag{1}$$

Stephen et al. [2] and Robertson and Zaragoza [26] present BM25F as an extension of the baseline BM25 [27] scoring function that is adapted to score field documents. Using the BM25F scheme presented in [28], an element score is computed as follows:

$$\text{Score}\,(e,q) = \sum_{t\in q\cup e} \frac{tf_{e,t}}{K + tf_{e,t}} * W_t, \tag{2}$$

where $\text{Score}(e,q)$ measures the relevance of element e to query q, $tf_{e,f}$ is a weighted normalized term frequency, K is a common tuning parameter for the BM25, and W_t is the inverse document frequency weight of term t.

The weighted normalized term frequency is obtained by first performing length normalization on term frequency $W_{e,f,t}$ of term t in field f in element e as follows:

$$W_{e,f,t} = \frac{tf_{e,f,t}}{1 + B_f * \left\langle \left(\text{len}_{e,f}/\text{avglen}_f\right) - 1 \right\rangle}, \tag{3}$$

where B_f is a smoothing parameter, $\text{len}_{e,f}$ is the length of field f, and avglen_f is the average length of elements in the entire collection after multiplying the normalized term frequency $W_{e,f,t}$ by field weight W_f:

$$tf_{e,t} = \sum_f W_f * W_{e,f,t}. \tag{4}$$

Kim and Croft [4] recently introduced the Field Relevance Model (FRM). FRM employs the notion of field relevance and a corresponding retrieval model between query terms and document fields, which are calculated by *Field Relevance* given a query $q = q_1, \ldots, q_m$, and *field relevance* $P(F_j \mid q_i, R)$ is the distribution of per-term relevance over document fields. *Field Relevance Model* is based on field relevance

estimates $P(F_j \mid q_i, R)$; the *Field Relevance Model* combines field-level scores $P(q_i \mid F_j, D)$ for each document using field relevance instead of weights as follows:

$$\begin{aligned}\text{Score}(e,q) = &\prod_{1\le i\le m}\sum_{1\le j\le n}\lambda P\left(F_j \mid q_i, R\right)\\ &+ (1-\lambda)\,P\left(q_i \mid F_j, D\right).\end{aligned} \tag{5}$$

Broschart and Schenkel [5] presented the use of proximity-aware scoring functions that lead to significant effectiveness improvements for XML retrieval. This method introduces modified proximity scores that take the document structure as follows:

$$\begin{aligned}\text{Score}(e,q) &= W_{t,e} + \text{Prox}_{t,e},\\ W_{t,e} &= \sum_{t\in q}\frac{(k_1+1)*tf_t}{K+tf_t} * ief_t,\\ ief_t &= \log\left[\frac{N-e_t+0.5}{e_t+1}\right].\end{aligned} \tag{6}$$

To compute the proximity part of the score for each term t, at first compute an accumulated score acc_t that depends on the distance of this term's occurrences in the element to other terms, adjacent query term occurrences using for each adjacent occurrence of a term t_j at distance d to an occurrence of t_i, the $\text{acc}_{t,i}$ grows by $(ief_t)/d$. The proximity score is computed as follows:

$$\text{Prox}_{t,e} = \sum_{t\in q}\min\{1, ief_t\}\,\frac{(k_1+1)*\text{acc}_t}{K+\text{acc}_t}, \tag{7}$$

where $\text{Score}(e,q)$ measures the relevance of element e to a query q, acct_t is calculated by $(ief_t)/d$.

Ogilvie and Callan [7] is based on language models and employs element-based indexing. Given a query q, terms t_i for each element e and its corresponding element language model $\ominus_e$, the element e is ranked as follows:

$$P(e \mid q) = P(e) * P\left(q \mid \ominus_e\right), \tag{8}$$

where $P(e)$ is the probability of relevance for element e and $P(q \mid \ominus_e)$ is the probability of the query q generated by language model $\ominus_e$. For instance,

$$P\left(t_1,\ldots,t_n \mid \ominus_e\right) = \prod_{i=1}^{n}\lambda P\left(t_i \mid e\right) + (1-\lambda)\,P\left(t_i \mid C\right), \tag{9}$$

where $P(t_i \mid e)$ is estimation of term t_i in element e, $P(t_i \mid C)$ is the probability of term t_i in collection C, and λ is the smoothing parameter.

To account for the length of an element e, and in particular for the heavily biased distribution of small elements in XML documents, which can be used to set $P(e)$ as follows [8]:

$$P(e) = \frac{\text{length}_e}{\sum_C \text{length}_e}, \tag{10}$$

where length_e is the length of element e and $\sum_C \text{length}_e$ is the length of element e occurring in collection C.

Theobald et al. [10] present the extended BM25 function in the TOPX, which is known as the Compactness of the baseline BM25 as follows:

$$\begin{aligned}&\text{Score}(e,q)\\ &= \sum_{t\in q\cup e}\frac{(k_1+1)*tf_{t,e}}{k_1*\left((1-b)+b*\left(\text{len}\left(e_A\right)/\text{avel}_A\right)\right)+tf_{t,e}}\\ &\quad * \log\frac{\left\langle\left(N_A-e_t+0.5\right)/e_t\right\rangle}{N_A+0.5},\end{aligned} \tag{11}$$

where $\text{len}(e_A)$ is the length of element e with tag A, avel_A is the average length of elements in the entire collection with tag A, k_1, and b is a common tuning parameter for the BM25.

The modified function provides a dampened influence of the $tf_{t,e}$ with tag A. However, this strategy is limited in that each tag name must be the same to implement automatic grouping and weight calculation.

The idea is to associate a weight to a structural constraint to reflect its significance. These weights are then used in the scoring function used to estimate an element relevance.With the increased availability of the data-centric a need for query in both structure and content of the XML documents has become explicit. As a result, a more complex information source is available, in fact, allowing us to improve the performance of search systems. Our approach considers the use of structure weight method, as discussed in Section 3.

3. Method

In this section, the search results become more refined at every step, and the refinement ultimately narrows down a set of potentially interesting documents. Below we describe our approach in more details.

3.1. Step 1: Elements Score. Firstly, we defined $\text{Score}(e, A = t)$ is a score for the relevance of a term t of an element e and then we used the baseline BM25 [27] in Sphinx (http://sphinxsearch.com/) [29] formula to score the element nodes according to query terms t contained in content conditions as follows:

$$\begin{aligned}&\text{Score}(e, A=t)\\ &= W_t * \frac{(k_1+1)*tf_{t,e}}{k_1*\left\langle(1-b)+b*(\text{len}(e)/\text{avel})+tf_{t,e}\right\rangle},\end{aligned} \tag{12}$$

where $\text{Score}(e, A = t)$ measures the relevance of element e to query term t, $tf_{t,e}$ is the frequency of term t occurring in element e, $\text{len}(e)$ is the length of element e, avel is the average length of elements in the entire collection, and k_1 and b are used to balance the weight of term frequency and element length.

And then, we compute the inverse element frequency W_t as follows:

$$W_t = \frac{\log\left\langle\left(N-e_t+1\right)/e_t\right\rangle}{\log(N+1)}, \tag{13}$$

Figure 3: Variation in the value of base a^n parameter.

where W_t is the inverse element frequency weight of term t, N is the total number of an element in the entire collection, and e_t is the total element of a term t occur.

For an "about()" function in NEXI operator with multiple terms that appeared to an element e, the aggregated score of e is simply computed as the sum of the element's scores for each term $t_1, \ldots, t_n$ conditions as follows:

$$\begin{aligned} \text{Score}(e,q) &= \text{Score}(e, A[\text{about}(t_1,\ldots,t_n)]) \\ &= \sum_{t_i \in n} \text{Score}(e, A = t_i). \end{aligned} \tag{14}$$

3.2. Step 2: Score Sharing Function. In the second step of our approach [30], we compute the scores of all elements from (14), in the collection that contains query terms. We consider the scores of elements e by accounting for their relevant descendants e_c. The scores of retrieved elements Score(e,q) are now shared between the leaf node and their parents in the document XML tree according to the following scheme:

$$\text{Score}(e,q) \longleftarrow \text{Score}(e,q) + \left\langle \sum_{e,c} \text{Score}(e_c,q) * \beta^n \right\rangle, \tag{15}$$

where Score(e,q) is a current parent node, Score(e_c,q) is a relevant child of element e, and β is a tuning parameter.

IF $\{0-1\}$ *THEN* preference is given to the leaf node over the parents.

OTHERWISE, preference is given to the parents.

n is the distance between the current parent node and the leaf node.

3.3. Step 3: Exponentiation Weight Function. The third step of our approach is the structure score evaluation. To improve the search result with richly structured, we assume that a query is composed of content (keywords) and structure constraints. The document-query similarity is evaluated by considering content and structure separately. We then combine these scores to the set of possibly relevant elements. Our structural scoring model essentially counts the number of navigational (i.e., element name-only) query conditions that are satisfied by a result candidate and thus considering the content conditions matched for the user queries. It assigns c_e for every directional condition that matched the element name $e_{\text{name}} \in d_{\text{path}}$ (i.e., an absolute path on the document structure). We analysed the structure for each topic in INEX as shown in Table 1 with respect to the INEX content-and-structure queries and each topic is including a few structure indications. Thus, we are proposed the novel of structural scoring when the user query is matching the structural constraints against the document tree using the *Exponentiation* is a^n.

Table 1: Report for structure in CAS topics of INEX.

CAS topics	Max.	Min.	Avg.
INEX-2006	5	1	**2.65**
INEX-2007	6	1	**2.87**
INEX-2008	4	1	**2.68**
INEX-2009	5	1	**2.47**
INEX-2010	4	1	**2.57**
INEX-2011	11	1	**2.75**

The bold font refer to the % that use to calculate value of improvement.

In order to evaluate the sensitivity of the *Exponentiation*, we have variation in the value of a parameter, including *base 10*, *base e*, *base 2*, *and base 1/2* as shown in Figure 3. According to the trend of the graph more smooth than other values and the powers of 2 are important in computer science because there are 2^n possible values for an n-bit binary variable. Thus, we simply for our algorithm calculate base on 2^{c_e}. After that we recomputed the element score Score(e,q) as follows:

$$\begin{aligned} c_e &= \sum_{e_{\text{name}} \in d_{\text{path}}} e_{\text{name}}, \\ \text{Score}(e,q) &\longleftarrow \text{Score}(e,q) * \langle 2^{c_e} \rangle, \end{aligned} \tag{16}$$

where c_e is the frequency of navigational condition that is matched with the $e_{\text{name}} \in d_{\text{path}}$.

In the following, we define $T(d)$ as the set of all elements in d that match the target element of the query. In document mode, every document d inherits the aggregated score among all target elements e, and these document scores Score(d,q) determine the output ranking among documents as follows:

$$\text{Score}(d,q) = \sum_{e \in T(d)} \text{Score}(e,q). \tag{17}$$

Table 2: The sphinx search modes.

Mode	Description
Match any	The final weight is a sum of weighted phrase ranks for matching *any* of the query words.
Match phrase	The final weight is the sum of weighted phrase ranks for matching the query *phrase*, which requires a perfect match.
Match extended	The final weight is the sum of weighted phrase ranks and the *BM25* weight, multiplied by a thousand and rounded to the nearest integer.

Table 3: The details of experiments.

Run	Exponentiation	Score Sharing
p16-BM25-EXPO	Yes	No
p16-TF-EXPO	Yes	No
p16-PHRASE-EXPO	Yes	No
p16-BM25	No	No
p16-TF	No	No
p16-PHRASE	No	No
p16-BM25-SS	No	Yes
p16-TF-SS	No	Yes
p16-PHRASE-SS	No	Yes

Table 4: Compare performing runs based on MAP with and without the exponentiation.

Run	MAP	P@10	P@20	P@30
p16-BM25-EXPO	0.3479	0.4316	0.3645	0.3298
p16-TF-EXPO	0.2125	0.2500	0.2171	0.1930
p16-PHRASE-EXPO	0.1937	0.2342	0.1921	0.1675
p16-BM25	0.1830	0.2184	0.1974	0.1939
p16-TF	0.0857	0.1447	0.1118	0.0921
p16-PHRASE	0.0857	0.1447	0.1118	0.0921
%	**52.60**	**50.60**	**54.16**	**58.79**

The bold font refer to the % that use to calculate value of improvement.

To see how users use structure in their queries, for instance, the user query needs "retrieve document sections with the paragraph p contains *xml retrieval*" as follows:

//section[about(//p, "xml retrieval")]

The first filter looks for occurrences of the term "xml" and "retrieval" in elements e whose context matches the path "//section//p" on the d_{path}. It is possible to assigning more weight for the return element e. In this case, we assume the Score(e, q) for each element e is 10, β is 0.7 and then the calculations are shown in Figure 4.

Thus, the Score(d, q) for the document d is $\langle 40 + 20 + 10 + 28 + 9.8 + 13.86\rangle = 121.66$.

Table 5: Compare performing runs based on MAP with and without the score sharing.

Run	MAP	P@10	P@20	P@30
p16-BM25-EXPO	0.3479	0.4316	0.3645	0.3298
p16-TF-EXPO	0.2125	0.2500	0.2171	0.1930
p16-PHRASE-EXPO	0.1937	0.2342	0.1921	0.1675
p16-BM25-SS	0.0641	0.0737	0.0908	0.1061
p16-TF-SS	0.0641	0.0711	0.0882	0.1044
p16-PHRASE-SS	0.0606	0.0605	0.0829	0.1070
%	**81.58**	**82.92**	**75.09**	**67.83**

The bold font refer to the % that use to calculate value of improvement.

Table 6: The significance (P) is computed with a 2-tailed t-test at MAP.

	Run	MAP
BM25	p16-BM25-EXPO	0.3479
	p16-BM25	0.1830
	P (t-test)	**0.48**
Score Sharing	p16-BM25-EXPO	0.3479
	p16-BM25-SS	0.0641
	P (t-test)	**0.75**

The bold font refer to the % that use to calculate value of improvement.

4. Experiment Setup

In this section, we present and discuss the results based on the INEX collection. This experiment was performed on Intel Pentium i5 4 ∗ 2.79 GHz with 6 GB of memory, Microsoft Windows 7 Ultimate 64 bit Operating System and Microsoft Visual C♯.NET 2008.

4.1. INEX Collection. The INEX-IMDB collection used in INEX 2010 (https://inex.mmci.uni-saarland.de) was generated from the plain text files published on the IMDB web site on April 10, 2010. There are two kinds of objects in the collection, movies and persons involved in movies. Each object is richly structured. For example, each movie has title, rating, directors, actors, and so forth; each person has name, birth date, and so forth. In total, the IMDB data collection contains 4,418,081 XML documents, including 1,594,513 movies, 1,872,471 actors, 129,137 directors who did not act in any movie, 178,117 producers who did not direct or act in any movie, and 643,843 other people involved in movies who did not produce or direct or act in any movie.

4.2. INEX Evaluations. The effectiveness of the retrieval results will be evaluated using the metrics as that in traditional IR, for example, precision, recall, MAP, P@10, P@20, and P@30 [31, 32]. Given a topic T and a set of documents D, each tested IR system returns an ordered subset $S = s_1, \ldots, s_n$ of D, ranked by the system's estimate of the likelihood that each document is relevant to T. Several effectiveness

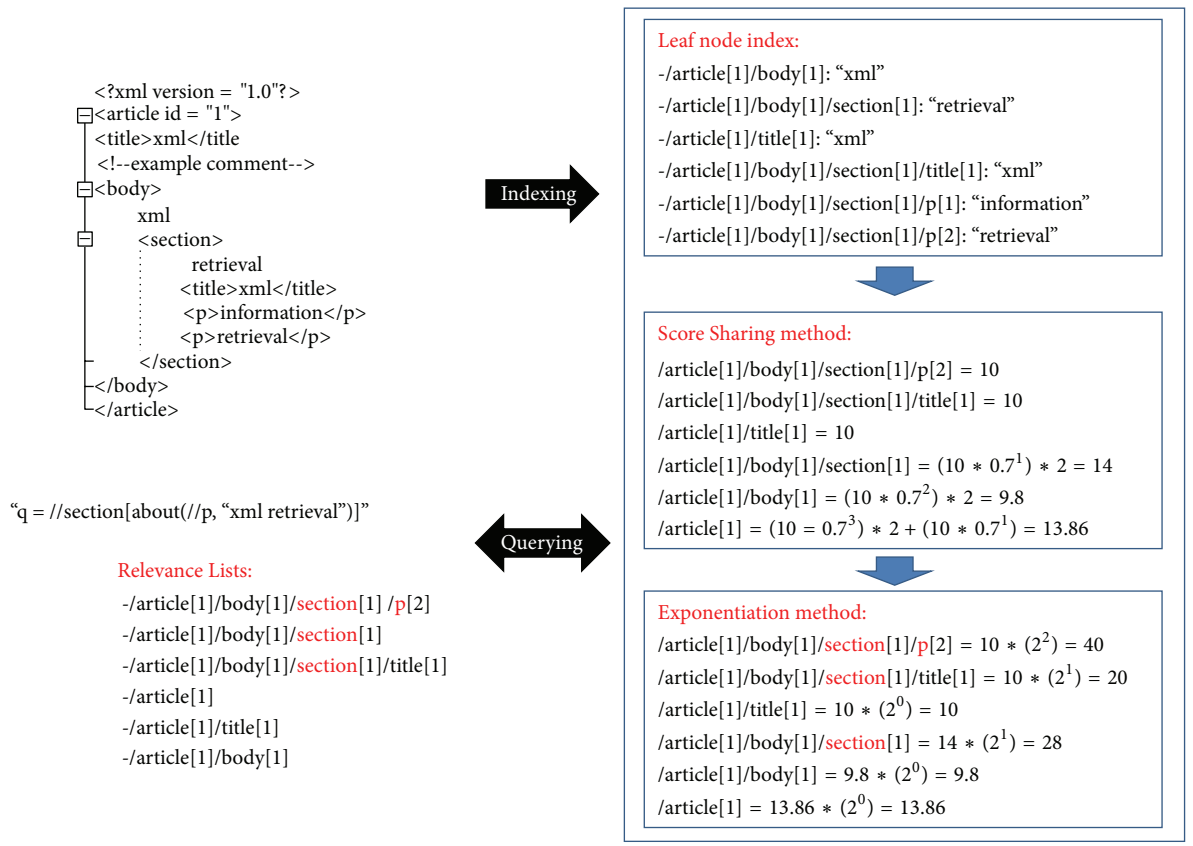

Figure 4: An Example of Exponentiation processing.

Table 7: Best performing runs based on MAP over the information topics.

Run	MAP	1/Rank	P@10	P@20
p16-BM25-EXPO [16]	**0.3564**	**0.8000**	**0.5000**	**0.4200**
p30-2011CUTxRun2 [17]	0.3449	0.7067	0.5000	0.4700
p47-FCC-BUAP-R1 [18]	0.3219	1.0000	0.5600	0.4300
p2-ruc11AMS [19]	0.3189	0.6500	0.4200	0.4500
p4-UAms2011adhoc [20]	0.3079	0.6750	0.3800	0.3100
p18-UPFbaseCO2i015 [21]	0.2576	0.6346	0.4600	0.4400
p77-PKUSIGMA02CLOUD [17]	0.2118	0.5015	0.4400	0.4200
p48-MPII-TOPX-20-co [17]	0.0900	0.3890	0.2600	0.1800
p12-IRIT-focus-mergeddtd-04 [22]	0.0366	0.3022	0.2200	0.1100

The bold font refer to the % that use to calculate value of improvement.

Table 8: Best performing runs based on 1/Rank over the known-item topics.

Run	MAP	1/Rank	P@10	P@20
p4-UAms2011adhoc	0.8112	0.9167	0.3167	0.2417
p2-ruc11AS2	0.7264	0.9167	0.3167	0.2417
p48-MPII-TOPX-20-co	0.2916	0.7222	0.2333	0.1833
p18-UPFbaseCO2i015	0.3752	0.7104	0.2500	0.2083
p16-BM25-EXPO	**0.4745**	**0.6667**	**0.0833**	**0.0417**
p77-PKUSIGMA01CLOUD	0.5492	0.6389	0.3167	0.2417
p30-2011CUTxRun2	0.3100	0.5730	0.2667	0.1750
p47-FCC-BUAP-R1	0.2500	0.3333	0.0333	0.0167
p12-IRIT-large-nodtd-06	0.0221	0.0487	0.0167	0.0333

The bold font refer to the % that use to calculate value of improvement.

Table 9: Best performing runs based on MAP over the list topics.

Run	MAP	1/Rank	P@10	P@20
p16-BM25-EXPO	**0.4251**	**0.7778**	**0.4778**	**0.3833**
p4-UAms2011adhoc	0.3454	0.6674	0.4222	0.3500
p77-PKUSIGMA02CLOUD	0.3332	0.5432	0.3889	0.3667
p2-ruc11AS2	0.3264	0.6488	0.4111	0.3333
p48-MPII-TOPX-20-co	0.2578	0.4926	0.3000	0.3333
p18-UPFbaseCO2i015	0.2242	0.5756	0.3556	0.3278
p12-IRIT-focus-mergeddtd-04	0.1532	0.2542	0.2333	0.2111
p30-2011CUTxRun3	0.0847	0.5027	0.1889	0.1611
p47-FCC-BUAP-R1	0.0798	0.3902	0.2889	0.2500

The bold font refer to the % that use to calculate value of improvement.

measures are computed, including average precision (AP); precision at k returned documents (P@k) defined as follows:

$$\mathrm{AP} = \frac{\sum_{k=1}^{|S|} \mathrm{rel}\left(s_k\right) * \mathrm{P@}k}{R},$$

$$\mathrm{P@}k = \frac{\sum_{i=1}^{k} \mathrm{rel}\left(s_i\right)}{k}, \tag{18}$$

$$R = \sum_{d_i \in D} \mathrm{rel}\left(d\right).$$

Performance across a set of topics is measured by calculating the mean of the AP values obtained by the measure for each individual topic, resulting in MAP. Assuming there are n topics:

$$\mathrm{MAP} = \frac{1}{n} * \sum_{t=1}^{n} \mathrm{AP}_t. \tag{19}$$

4.3. Results and Discussion. In this section, we tuned the β parameter using INEX-2005 ad hoc track evaluation scripts distributed by the INEX organizers. Our tuning approach was such that the sums of all relevance scores are maximized and then the total number of leaf node is 2500 and the β parameter is set to 0.60. Following that, we used the Sphinx parameters for the BM25 where $k_1 = 1.20$ and $b = 0.00$ and the entire Sphinx match mode values in our experiment include MATCH ANY (TF), MATCH PHRASE (PHRASE), and MATCH EXTENDED (BM25) and are provided in Table 2. The main components of the MEXIR [33] retrieval system are as follows.

(1) When new documents are entered into the system, the Absolute Document XPath Indexing (ADXPI) [34] indexer parses and analyzes the name of each element and its position to build inverted lists for each index in this system.

(2) The SphinxDB search engine is used to build both indices in the system. The Selected Weight index is based on term frequency, and the Leaf Node index is based on the classic BM25 function.

(3) The Score Sharing function is used to assign parent scores by assigning a proportion of the scores of the leaf nodes to their parents using a top-down approach.

(4) The Exponentiation function is used to adjust the element scores based on linear combination.

The MEXIR search engine retrieves XML elements based on the leaf node indexed with respect to the significant words including the Exponentiation and Score Sharing functions, and then we combine relevance score from the element into the document score. Thus, the document with the higher relevance score will be chosen as the retrieval set. The details of experiment are shown in Table 3.

The performance of different features and ranking methods can now be evaluated. In order to deepen into the analysis of the Exponentiation scoring function, we have also run experiments to study the impact of structure weight with the content-and-structure query in the performance. Table 4 shows the results compared for the best performing runs with and without Exponentiation technique. The **p16-BM25-EXPO** used the Exponentiation for boosting element score, and the **p16-BM25** is the baseline BM25 and then the Exponentiation function was shown to improve the effectiveness of search system measured in terms of MAP, P@10, P@20, and P@30 and are 52.60%, 50.60%, 54.16%, and 58.79%, respectively. Table 5 shows the results compared for the best performing runs with and without the Score Sharing technique. The **p16-BM25-EXPO** is used the Exponentiation and the used the Score Sharing is the **p16-SS-SW** and then the Exponentiation weight shown improve the effectiveness of over the Score Sharing technique measured in terms of MAP, P@10, P@20 and P@30 are 81.58%, 82.92%, 75.09% and 67.83%, respectively. It can be seen, that **p16-BM25-EXPO** obtained the best performance, although the improvement over both the baseline BM25 and the Score Sharing is significant for most of the considered metrics. The significance (P) was computed with a 2-tailed t-test as shown in Table 6. The **p16-BM25-EXPO** improved by 0.48% over the baseline BM25 at MAP, and 0.75% over the baseline BM25 with the Score Sharing at MAP on INEX-IMDB collection.

In this analysis, we take the results that were obtained from BM25 over the Exponentiation and compare them with the results from the baseline BM25 and over the Score Sharing function. It is shown again that Exponentiation

works well with the document-centric XML documents. We can conclude that significant improvement of results of the Exponentiation function can be obtained from the content-and-structure query and document structure. This finding suggests that it is possible to improve the TF, PHRASE, and the baseline BM25 approaches, which are the usual benchmarks in INEX. The main conclusion that can be drawn from the experiments is that the Exponentiation function is successful in structure weight and could be utilized to improve the effectiveness of search systems.

Another major conclusion, is that we analyzed the effectiveness of the runs for each of the three topic types with respect to the INEX [17] and the results are presented in Tables 7, 8, and 9. The overall results are satisfactory if we compare them with those obtained by participants in the INEX contests. On comparing the effectiveness for the informational topics, our run ranked first, scoring 0.3564, measured with MAP; it ranked fifth scoring 0.6667, measured with 1/Rank for the known-item topics; and in the results of the list topics, our run ranked first, scoring 0.4251, measured with MAP.

In this analysis, we take the results that were obtained from the INEX report [17]. It is shown again that our system works well with the *List* and *Informational* topics of the document-centric XML documents measured with the MAP metric. Unfortunately, on the known-item topics, the relevant answer is a single document; in this area, the performance was not satisfactory and so further investigation is required.

5. Conclusions

With the increased availability of the data-centric a need for query in both structure and content of the XML documents has become explicit. As a result, a more complex information source is available, in fact, allowing us to improve the performance of search systems. In this paper, we are investigating retrieval techniques and related issues over a strongly structured collection using the Exponentiation weight for the document's structure over the content-and-structure query, in the data-centric track of the INEX 2011. Our expectation is that structure weighted will improve the effectiveness of the search systems. In terms of processing time, our system required an average of one second per topic. In addition, our run for the ad hoc task showed that the structural information could be utilized to improve the effectiveness of the search system over the baseline BM25 measured in terms of MAP, P@10, P@20, and P@30 and are 52.60%, 50.60%, 54.16%, and 58.79% and over the Score Sharing technique measured in terms of MAP, P@10, P@20, and P@30 and are 81.58%, 82.92%, 75.09%, and 67.83%, respectively. The success of our ad hoc run indicates that indexing the complete XML structure of IMDB and the structure weights are necessary for effective document retrieval in the search system.

In future work, we will look closer at the relative value of various types of metadata, tags, and subject headings. We will also look at the different weighting methods underlying the relevance judgements and topic categories, such as blind feedback and recommendation search.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] J. Kamps, M. Marx, M. de Rijke, and B. Sigurbjornsson, "Structured queries in xml retrieval," in *Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM '05)*, pp. 4–11, ACM, 2005.

[2] R. Stephen, Z. Hugo, and T. Michael, "Simple bm25 extension to multiple weighted fields," in *Proceedings of the 13th ACM International Conference on Information and Knowledge Management (CIKM '04)*, D. Grossman, L. Gravano, C. Zhai, O. Herzog, and D. A. Evans, Eds., pp. 42–49, ACM Press, 2004.

[3] J. Kim, X. Xue, and W. B. Croft, "A probabilistic retrieval model for semistructured data," in *Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval (ECIR '09)*, pp. 228–239, Springer, 2009.

[4] J. Y. Kim and W. B. Croft, "A field relevance model for structured document retrieval," in *Proceedings of the 34th European conference on Advances in Information Retrieval (ECIR '12)*, pp. 97–108, Springer, 2012.

[5] A. Broschart and R. Schenkel, "Proximity-aware scoring for XML retrieval," in *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '08)*, pp. 845–846, July 2008.

[6] T. Schlieder and H. Meuss, "Querying and ranking XML documents," *Journal of the American Society for Information Science and Technology*, vol. 53, no. 6, pp. 489–503, 2002.

[7] P. Ogilvie and J. Callan, "Language models and structured document retrieval," in *Proceedings of the 1st Workshop of the INitiative for the Evaluation of XML Retrieval (INEX '03)*, pp. 18–23, Dagstuhl, Germany, 2003.

[8] J. Kamps, M. De Rijke, and B. Sigurbjörnsson, "The importance of length normalization for XML retrieval," *Information Retrieval*, vol. 8, no. 4, pp. 631–654, 2005.

[9] A. Theobald and G. Weikum, "The index-based xxl search engine for querying xml data with relevance ranking," in *Proceedings of the 8th International Conference on Extending Database Technology: Advances in Database Technology*, pp. 477–495, Springer, 2002.

[10] M. Theobald, H. Bast, D. Majumdar, R. Schenkel, and G. Weikum, "TopX: efficient and versatile top-k query processing for semistructured data," *VLDB Journal*, vol. 17, no. 1, pp. 81–115, 2008.

[11] M. Gery, C. Largeron, and F. Thollard, "Ujm at inex 2008: pre impacting of tags weights," in *Proceedings of the 7th International Workshop of the Initiative for the Evaluation of XML Advances in Focused Retrieval*, Lecture Notes in Computer Science, pp. 46–53, Springer, 2008.

[12] A. Trotman and M. Lalmas, "Why structural hints in queires do not help XML-retrieval," in *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 711–712, ACM, August 2006.

[13] S. Geva, "Gpx—gardens point xml information retrieval at inex 2006," in *Proceedings of the 5th International Workshop of the Initiative for the Evaluation of XML Comparative Evaluation of XML Information Retrieval Systems*, Lecture Notes in Computer Science, pp. 137–150, Springer, 2007.

[14] P. Ogilvie and J. Callan, "Parameter estimation for a simple hierarchical generative model for xml retrieval," in *Proceedings of the 4th International Workshop of the Initiative for the Evaluation of XML Advances in XML Information Retrieval and Evaluation*, Lecture Notes in Computer Science, pp. 211–224, Springer, 2006.

[15] Y. Mass and M. Mandelbrod, "Using the inex environment as a test bed for various user models for xml retrieval," in *Proceedings of the 4th International Workshop of the Initiative for the Evaluation of XML Advances in XML Information Retrieval and Evaluation*, Lecture Notes in Computer Science, pp. 187–195, Springer, 2006.

[16] T. Wichaiwong and C. Jaruskulchai, "Mexir at inex-2011," in *Proceedings of the 10th International Workshop of the Initiative for the Evaluation of XML Focused Retrieval of Content and Structure*, Lecture Notes in Computer Science, pp. 180–187, Springer, 2012.

[17] M. Theobald, Q. Wang, G. Ramrez, M. M. Marx, M. Theobald, and J. Kamps, "Overview of the inex 2011 data-centric track," in *Proceedings of the 10th International Workshop of the Initiative for the Evaluation of XML Retrieval Focused Retrieval of Content and Structure (INEX '12)*, Springer, 2012.

[18] D. V. Ayala, D. Pinto, S. L. Silverio, E. Castillo, and M. T. Vidal, "Buap: a recursive approach to the data-centric track of inex 2011," in *Proceedings of the 10th International Workshop of the Initiative for the Evaluation of XML Focused Retrieval of Content and Structure*, Lecture Notes in Computer Science, pp. 161–166, Springer, 2012.

[19] Q. Wang, Y. Gan, and Y. Sun, "Ruc at inex 2011 data-centric track," in *Proceedings of the 10th International Workshop of the Initiative for the Evaluation of XML Focused Retrieval of Content and Structure*, Lecture Notes in Computer Science, pp. 167–179, Springer, 2012.

[20] A. Schuth and M. Marx, "University of amsterdam data centric ad hoc and faceted search runs," in *Proceedings of the 10th International Workshop of the Initiative for the Evaluation of XML Focused Retrieval of Content and Structure*, Lecture Notes in Computer Science, pp. 155–160, Springer, 2012.

[21] G. Ramrez, "Upf at inex 2011: books and social search track and data-centric track," in *Proceedings of the 10th International Workshop of the Initiative for the Evaluation of XML Focused Retrieval of Content and Structure*, Lecture Notes in Computer Science, pp. 146–154, Springer, 2012.

[22] C. Laitang, K. Pinel-Sauvagnat, and M. Boughanem, "Edit distance for xml information retrieval: some experiments on the datacentric track of inex 2011," in *Proceedings of the 10th International Workshop of the Initiative for the Evaluation of XML Focused Retrieval of Content and Structure*, Lecture Notes in Computer Science, pp. 138–145, Springer, 2012.

[23] A. Trotman and M. Lalmas, "The interpretation of cas," in *Proceedings of the 4th International Workshop of the Initiative for the Evaluation of XML Advances in XML Information Retrieval and Evaluation*, Lecture Notes in Computer Science, pp. 58–71, Springer, 2005.

[24] A. Trotman and B. Sigurbjörnsson, "Nex, now and next," in *Proceedings of the 3rd International Workshop of the Initiative for the Evaluation of XML Advances in XML Information Retrieval*, vol. 3493 of *Lecture Notes in Computer Science*, pp. 41–53, Springer, 2004.

[25] A. Trotman and B. Sigurbjörnsson, "Narrowed extended xpath i (nexi)," in *Proceedings of the 3rd International Workshop of the Initiative for the Evaluation of XML Advances in XML Information Retrieval*, vol. 3493 of *Lecture Notes in Computer Science*, pp. 16–40, Springer, 2004.

[26] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: BM25 and beyond," *Foundations and Trends in Information Retrieval*, vol. 3, no. 4, pp. 333–389, 2009.

[27] R. Stephen, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford, "Okapi at trec-3," in *Overview of the Third Text Retrieval Conference*, pp. 109–126, National Institute of Standards and Technology, 1994.

[28] K. Y. Itakura and C. L. A. Clarke, "A framework for BM25F-based XML retrieval," in *Proceedings of the 33rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '10)*, pp. 843–844, July 2010.

[29] A. Aksyono, *Introduction to Search with Sphinx*, O'Reilly Media, 2011.

[30] T. Wichaiwong and C. Juruskulchai, "A score sharing method for xml element retrieval information," *An International Interdisciplinary Journal*, vol. 15, no. 10, pp. 4165–4178, 2012.

[31] R. A. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval, The Concepts and Technology Behind Search*, Addison Wesley Longman, Boston, Mass, USA, 2011.

[32] J. Kamps, J. Pehcevski, G. Kazai, M. Lalmas, and S. Robertson, "Inex 2007 evaluation measures," in *Proceedings of the 6th International Workshop of the Initiative for the Evaluation of XML Focused Access to XML Documents*, Lecture Notes in Computer Science, pp. 24–33, Springer, 2007.

[33] T. Wichaiwong and C. Jaruskulchai, "Mexir: an implementation of high performance and high precision on xml retrieval," *Computer Technology and Application*, vol. 2, no. 4, pp. 301–310, 2011.

[34] T. Wichaiwong and C. Jaruskulchai, "XML retrieval more efficient using ADXPI indexing scheme," in *Proceedings of the 25th IEEE International Conference on Advanced Information Networking and Applications Workshops (WAINA '11)*, pp. 638–643, March 2011.