International Journal of Web Information Systems

Facet-value extraction scheme from textual contents in XML data

Takahiro Komamizu Toshiyuki Amagasa Hiroyuki Kitagawa

Received 6 April 2015
Revised 6 April 2015
Accepted 17 April 2015

Facet-value extraction scheme from textual contents in XML data

Takahiro Komamizu
Center for Computational Sciences, University of Tsukuba, Tsukuba, Japan, and
Toshiyuki Amagasa and Hiroyuki Kitagawa
Faculty of Engineering, Information and Systems, University of Tsukuba, Tsukuba, Japan

Abstract

Purpose – The purpose of this paper is to extract appropriate terms to summarize the current results in terms of the contents of textual facets. Faceted search on XML data helps users find necessary information from XML data by giving attribute–content pairs (called facet-value pair) about the current search results. However, if most of the contents of a facet have longer texts in average (such facets are called textual facets), it is not easy to overview the current results.

Design/methodology/approach – The proposed approach is based upon subsumption relationships of terms among the contents of a facet. The subsumption relationship can be extracted using co-occurrences of terms among a number of documents (in this paper, a content of a facet is considered as a document). Subsumption relationships compose hierarchies, and the authors utilize the hierarchies to extract facet-values from textual facets. In the faceted search context, users have ambiguous search demands, they expect broader terms. Thus, we extract high-level terms in the hierarchies as facet-values.

Findings – The main findings of this paper are the extracted terms improve users' search experiences, especially in cases when the search demands are ambiguous.

Originality/value – An originality of this paper is the way to utilize the textual contents of XML data for improving users' search experiences on faceted search. The other originality is how to design the tasks to evaluate exploratory search like faceted search.

Keywords Managing and storing XML data, Applications of Web mining and searching, Indexing and retrieval of XML data

Paper type Research paper

1. Introduction

XML (Extensible Markup Language) (W3C, 2015) has become a *de facto* standard data format for textual data with complex structure and has been used in many applications. XML data consist of texts and nested tags over them, thereby allowing users to represent complex data structure. From the nested structure, XML data can be represented as tree structure (an example is shown in Figure 1). Because of the simplicity and versatility of XML, it has been used as standardized data exchange format, e.g. RDF/XML[1] and SOAP[2]. Also, XML has been used in many applications from various domains: examples in chemical domain, including Swiss-Prot[3] and KEGG[4], and

This research was partly supported by the Grant-in-Aid for Scientific Research (B) (#26280037) and the program Research and Development on Real World Big Data Integration and Analysis of the Ministry of Education, Culture, Sports, Science and Technology, Japan.

International Journal of Web Information Systems
Vol. 11 No. 3, 2015
pp. 270-290
© Emerald Group Publishing Limited
1744-0084
DOI 10.1108/IJWIS-04-2015-0012

those in business applications, including ebXML[5] and XBRL[6]. In addition, Wikipedia and bibliography database DBLP support XML data as a download format.

There are roughly two kinds of subtree search methods for XML, namely, path-based and keyword-based method. The path-based method is based on XPath; a user writes an XPath query specifying the location of desired XML elements in the XML data. A full-fledged query language, XQuery, is also available for path-based search. Besides, keyword-based method is used, in which a user gives a set of keywords and gets the result subtrees containing all keywords. To achieve this, lowest common ancestor (LCA)-based approaches have been proposed (Li *et al.*, 2004; Xu and Papakonstantinou, 2005; Li *et al.*, 2007), which return the lowest subtrees containing all keywords. The aforementioned methods give appropriate search results when users have concrete search demands. On the contrary, when users do not have such concrete search demands, these methods return large number of search results, and users are required to modify query expression or to look into the search results to find desired information. To deal with such situations, exploration over search results is an important approach for users to obtain desired results.

Faceted search (Tunkelang, 2009) is one of the exploratory search methods (White *et al.*, 2006), which enables users to explore desired information in an interactive manner. In faceted search, faceted search system shows users an overview of the current search results in terms of facets. Taking the bibliographic record database as an example, results of this dataset are papers, and the facets for the results are author, title, venue and year. By seeing the facets and their values, a user restricts the current search results by selecting a facet and its value. For example, suppose the user selects a value "2014" on the year facet, the faceted search system returns the results restricted to papers published in 2014. Again, the user can see possible restriction conditions from the newly calculated facets and their values for the current results. This interactive nature of the faceted search has been accepted by many users, so it has become popular, especially for search interfaces for databases containing a large number of data. There are many real applications which utilize faceted search interface, e.g. DBLP[7] and eBay[8]. As faceted search is compatible with keyword search, such real applications also combine keyword search interface with faceted search interface.

Applying faceted search for XML data is challenging, because XML does not have units of objects, whereas ordinary structured record data have. Hence, we need to define XML subtrees in XML data as objects in advance. Most of existing approaches define objects and facets manually. Consequently, it imposes a large burden on those who decide objects and facets. Our previously proposed framework (Komamizu *et al.*, 2011)

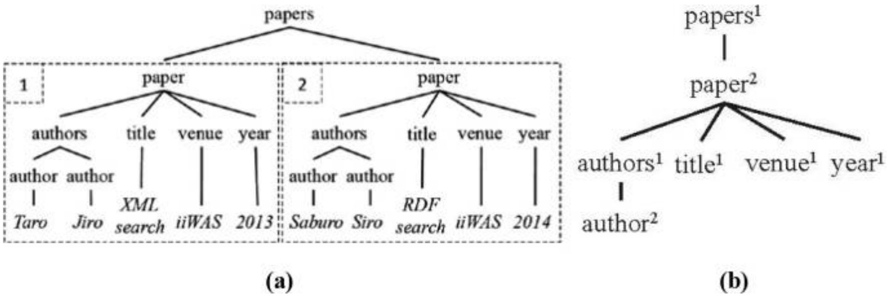


Figure 1.
Example of (a) XML data and (b) its structural information

reduces this burden by semi-automated extraction process of objects and facets. Moreover, in Komamizu *et al.* (2014) we automate extraction process by heuristic approaches. For the flexibility of search method, we make it possible to use keyword search on the framework using meaningful lowest common ancestor (MLCA) (Li *et al.*, 2004). The detail of this framework is going to be explained in Section 3.

In the framework, facets with long and unique textual values are removed by system managers (Komamizu *et al.*, 2011) or excluded from faceted search interface (Komamizu *et al.*, 2014), because these facets have unique values on each result subtrees like identifiers. We explicitly call facets with long and unique textual values as textual facets. A textual facet, each of which values identifies one result subtree, affects search performance of faceted search interface. This is because, if the user already knows the value of such facet, she has specific search demand in which path-based search and keyword-based methods work enough, and this case is not in our scope. For bibliographic dataset example, title is an example of textual facets, as titles of papers are almost unique. However, values of textual facets still contain informative contents such as topical terms. Hence, in this paper, we try to make the best use of textual facets.

In this paper, we propose a scheme to extract conceptual terms as values of textual facets from long text values in the context of faceted search for XML data by extending the faceted search framework for XML data (Komamizu *et al.*, 2011). Our approach is based on approaches which construct concept hierarchy from textual documents like subsumption algorithm (Sanderson and Croft, 1999). The goal of concept hierarchy construction approaches is to determine hypernym–hyponym relationship (a.k.a. subsumption) between two terms from large number of textual documents. As our goal is to extract term hierarchy which covers (most of) all result subtrees, we extend the subsumption algorithm considering coverage of terms over documents.

We also propose an evaluation methodology where we can vary the specificities of tasks. Evaluating exploratory search systems is difficult to guarantee reproductivity of the searching tasks over different datasets. We propose a concept specification level of tasks which indicate how a task is specific on limited number of results. The specification level is evaluated by selectivity of the terms in the task. In addition to the specification level, we give a template-based task designing strategy using specification level. We evaluate our proposed approach using a publicly available dataset by designing tasks changing the specification level. Also, we analyse stability of the specification level on the evaluation.

The contributions of this paper can be summarized as follows:

- We propose an extraction approach of values of textual facets, which is based on concept hierarchy constructions methods.
- We introduce an evaluation methodology of exploratory search using specification level and analyse stability of evaluation using specificity, as well as a task-designing methodology.
- Based on the evaluation methodology, we evaluate the proposed approach and the experimental results show that our proposed approach in this paper improves search performance, especially when specification level is low.

The rest of this paper is organized as follows: Section 2 explains related work, including our previous work (Komamizu *et al.*, 2011) to locate our research area among existing researches. Section 3 briefly introduces the basic framework on the previous work (Komamizu *et al.*, 2011), including basic definitions. Section 4 introduces the extraction mechanism of values of textual facets over XML data. Section 5 shows an evaluation methodology and the efficiency of our proposed approach, and Section 6 concludes this paper and describes future works.

2. Related work

Related work of this paper is categorized into the following four categories:

1. faceted search;
2. faceted search for semi-structured data like XML and RDF;
3. facet-value extraction from textual documents; and
4. experimental task design on exploratory search.

In the first category, we summarize current topics on faceted search related researches. Then, for the second category, we show related activities which enable faceted search for semi-structured data. For the third category, we introduce researches on facet-value extraction from textual documents. Finally, we show existing evaluation methods for exploratory search.

Faceted search has been well-studied for structured data. Yee *et al.* (2003) have proposed faceted search interface from attributed images. Each attribute on images is treated as facets. They succeeded in providing users a user interface to navigate images via facets. For structured data, facet ranking has been studied (Dash *et al.*, 2008; Roy *et al.*, 2008; Kashyap *et al.*, 2010) to show interesting facets for each faceted search results to navigate further. In such facet ranking methods, textual facets are ranked lower, even though such text values include informative contents.

With the success of faceted search, many researches apply faceted search for various kinds of data. Oren *et al.* (2006) have proposed faceted search for RDF data. They treat RDF nodes as objects and extract RDF predicates as facets and RDF node directed by the facet predicates as values of facets. Koren *et al.* (2007) show extensibility of faceted search system into a file system which treats tremendous number of files. Li *et al.* (2010) apply faceted search for exploration on the Wikipedia graph, and Wang *et al.* (2013) apply faceted search for program exploration. For those works, they assume that there are only short, meaningful and non-unique texts for facets. However, in many cases, there exist long and unique textual values which containing informative contents, like description about files, descriptive contents of DBPedia and comments of programs.

The closest research to our work is to apply faceted search for XML data. There exist few researches about this research area. Marwick (2008) has introduced faceted navigation for XML document. The approach of Marwick (2008) is rather straightforward; the definitions of target subtrees and facets are assumed to be predetermined, while respective values are extracted through XPath. Hence, Marwick (2008) needs to determine target subtrees and facets whenever a system manager wants to construct faceted search interface over XML data. Our previous work (Komamizu *et al.*, 2011; Komamizu *et al.*, 2014) has overcome (Marwick, 2008) by developing general framework to construct faceted search interface over XML data. In Komamizu *et al.* (2011), we extract candidate target subtrees and facets automatically, and then the system manager chooses feasible target subtrees and facets for the faceted search interface. Also, we automate extraction process of target subtrees and facets in Komamizu *et al.* (2014). To the best of our knowledge, no other researches directly working on faceted search for XML data exist.

In related work for methods of facet extraction from textual documents, there are roughly two subcategories: one is extraction from textual documents themselves (Sanderson and Croft, 1999; Pound *et al.*, 2011; Abel *et al.*, 2011; Kong and Allan, 2013) and the other is extraction using external knowledge (Stoica *et al.*, 2007; Hahn *et al.*, 2010). Sanderson and Croft (1999) have proposed subsumption algorithm. Subsumption algorithm computes whether term x subsumes another term y using conditional probability. Pound *et al.* (2011) extract facets for Web search from a number of query logs by computing co-occurrences of terms in the query logs, while Kong and Allan (2013) extract facets for Web search from result pages, by modeling facets of terms in the results pages by graphical model. In addition, Abel *et al.* (2011) extract facets from tweets. In Abel *et al.* (2011), tweets are considered as objects, and entities in the tweets are facet-value pairs (the types of the entities are facet and the names of entities are values of the facets). For the latter subcategory, Stoica *et al.* (2007) have proposed CastaNet algorithm to derive concept hierarchy from WordNet hypernyms and Hahn *et al.* (2010). The difference of our work in this paper comparing with subsumption (Sanderson and Croft, 1999) and CastaNet (Stoica *et al.*, 2007) is the presence of attributes other than texts. The presence affects the constructed hierarchies, as it changes the set of documents for the hierarchies.

Experimental evaluations on exploratory search is still challenging because of the instability of the evaluations, because they are depended on examinees. Yee *et al.* (2003) evaluate usability by questionnaires, including easiness and satisfiability, and the criterion for experimental task design is not clearly mentioned. Oren *et al.* (2006) conducts the similar experimental evaluation. Koren *et al.* (2008) perform experiments by counting number of actions, and Abel *et al.* (2011) follow this, but both of them do not mention about task design. Kules *et al.* (2009), give a set of characteristics of tasks and a template of task scenarios. We use this template of Kules *et al.* (2009) by extracting terms to put into the template (the detail is going to be explained in Section 5).

3. Framework of faceted search over XML data

This section explains the framework for faceted search over XML data proposed in our previous work (Komamizu *et al.*, 2011). The framework consists of two phases, namely, construction phase and retrieval phase. The construction phase extracts objects and facets from given XML data, and the retrieval phase navigates users to search objects using extracted facets.

To define objects, the framework utilizes structural information of given XML data, which is a summary of possible structure of the XML data (e.g. DataGuide) (Goldman and Widom, 1997). The structural information in this framework includes average occurrence, as frequency label of an XML element under its parental elements. Figure 1 shows an example of XML data of bibliographic information [Figure 1(a)] and its structural information [Figure 1(b)]. As is often the case with XML data, structural information of XML data are represented as tree structure. The XML data are rooted by papers element, and two paper elements are child elements. Thus, in the structural information, papers is the root and it has a paper element as its child element with the occurrence “2” as its label. Each paper element in the XML data contains authors, title, venue and year child elements so that paper element in the structural information has these elements as its child elements with the frequency label "1".

In this framework, frequently occurring elements are extracted as objects. In the structural information, an element with the frequency label greater than a frequency threshold θ tends to consist of a unit of data; thus, such elements are regarded as objects (Definition 1).

Definition 1 (object): Given XML data, its structural information, and a frequency threshold θ, an object is defined as an XML subtree rooted by the XML element which corresponding node in structural information having frequency label greater than θ. □

In the example of Figure 1, suppose that the frequency threshold is set to one, paper and author elements are considered as objects. However, author element has no child element, so, in this example, subtrees rooted by paper elements are extracted as objects (dashed boxes in Figure 1(a)).

Another essential to enable faceted search is to extract facets for the extracted objects. As there are several options to define facets for objects, like all elements below in the objects, leaf elements and more sophisticated ways like machine learning-based extractions. For the simplicity, we in this paper use Definition 2, formally.

Definition 2 (facet): Given XML data and its structural information, a facet of an object is a leaf node in the structural information which is one of descending nodes of the node corresponding to the object. A value of the facet on the object is a textual content of the XML element corresponding to the facet in the object. □

Facets for the paper objects extracted in the above example are author, title, venue and year elements. As is discussed in Komamizu *et al.* (2014), facets containing (almost) unique value for each object, like identifier, are excluded from faceted search interface due to its inefficiency as restriction conditions in the faceted search context. In this example, title facet is excluded, since titles of papers are almost unique in general. As a consequence, from the example, two paper objects are extracted, and author, venue and year are used as facets for the paper objects. Using these facets, a user can search objects by selecting facets. Suppose that the user selects year facet and its value "2014", and then the system returns a paper object numbered two as a result.

In addition, as faceted search lives well together with keyword search, the framework enables keyword search using MLCA (Li *et al.*, 2004) technique. For the input keywords, the keyword search engine returns objects matching with the input keywords. For example, suppose that a user gives a keyword "XML", the keyword search engine returns the paper object numbered one as a result.

4. Textual facet extraction

The previous works (Komamizu *et al.*, 2011; 2014) explained in the previous section makes faceted search for XML data easier to build the faceted search interface. In the framework, facets having unique values for each object are excluded by the system managers who are responsible for constructing faceted search interface (Komamizu *et al.*, 2011) or excluded automatically (Komamizu *et al.*, 2014). The uniqueness comes from various cases: the facets act as identifiers of objects and the facets contain long textual values. Consequently, each value becomes unique. Examples of the latter case include titles of papers and descriptions of products. Although the facets with long textual values are excluded so that they contain unique values, the facets still include informative contents in their values. The main objective of this paper is to extract facet-values from long and unique textual contents. We call such facets as textual facets and give definition of textual facet in Definition 3.

Definition 3 (textual facet): Given a length threshold α and a uniqueness threshold β, a facet is textual facet if the average length of all values of the facet is long than α and the average occurrence of each distinct value among whole objects is less than β.

A textual facet contains informative contents like topical terms. For example, a title of a paper in the bibliographic database (like DBLP) contains unique but long texts, as titles of papers are almost unique but they briefly explain about the contents of the papers. A concrete example of title is taken from our previous paper, "A Framework of Faceted Navigation for XML Data". This implies that this paper is about "faceted navigation" and "XML". Another example is a description about a movie in the movie database (like IMDB). It contains long text of introduction about the movie, and the descriptions about movies are almost unique because few movies are exactly same story and casts. As is shown in these examples, textual facets (title and description in these cases) contain informative contents like topical terms of papers and background, genres, casts, etc. about movies. Therefore, utilizing textual facets by extracting informative terms from textual contents of the facets is expected to improve search experience.

In the rest of this section, we introduce the basic idea of our approach in Section 4.1, our proposed methods in Section 4.2 and a prototypical faceted search interface in Section 4.3.

4.1 Basic idea

The basic idea of our approach is based on subsumption algorithm (Sanderson and Croft, 1999) which aims at constructing concept hierarchy from a number of textual documents. The subsumption algorithm computes the hypernym–hyponym relationships (or subsumption relationships) between two terms using conditional probability computed from occurrences of terms among the documents. The work (Dakka *et al.*, 2005) generalizes subsumption algorithm between two terms using threshold parameters, as Sanderson and Croft (1999) fixes the parameters experimentally. The subsumption relationship between two terms, say x and y, for given subsumption threshold τs and directionality threshold τd are computed as follows: x subsumes y if $p(x|y) > \tau s$ and $p(x|y) > \tau d \cdot p(y|x)$. Dakka *et al.* (2005) has experimented choices of thresholds, and it says that $\tau s = 0.8$ and $\tau d = 1.2$ have recorded as the best, so we in this paper use the same threshold values for subsumption algorithm.

We show an example of subsumption algorithm using the following document set: (Table I).

Suppose to check "XML" subsumes "search", from this table, we compute the conditional probabilities as:

$$p(\text{"XML"}\,|\,\text{"search"}) = 6/9 \approx 0.667$$
$$p(\text{"search"}\,|\,\text{"XML"}) = 5/6 \approx 0.833$$

Because of p ("XML" | "search") <Ts = 0.8, "XML" does not subsume "search". On the other hand, "search" subsumes "XML" because:

$$p(\text{"search"}\,|\,\text{"XML"}) > \text{Ts, and}$$
$$p(\text{"search"}\,|\,\text{"XML"}) > \text{Td} \cdot p(\text{"XML"}\,|\,\text{"search"}) = 0.8$$

ID	Text
1	XML search
2	Faceted XML search
3	XML keyword search
4	XML query suggestion
5	XML search log analysis
6	Indexing for XML keyword search
7	RDF search
8	Faceted RDF search
9	RDF keyword search

Table I.

Similarly, we can obtain the terms "search" subsumes as "XML", "RDF", "key-word" and "faceted". From this subsumption, we can observe various kinds of "search" that exists in the document set.

Algorithm 1 Iterative subsumption algorithm.
Input: a set of objects D, coverage threshold θ, thresholds for subsumption τ_s and τ_d
Output: A set of concept hierarchies H

```
$H \leftarrow \{\}, C \leftarrow \{\}, U \leftarrow D$
$rc \leftarrow |C|/|D|$
while $rc < \theta$ do
    $U \leftarrow D \backslash C$
    $h \leftarrow subsumption(U, \tau_s, \tau_d)$
    $C \leftarrow C \cup cover(D, h)$
    $H \leftarrow H \cup \{h\}$
    $rc \leftarrow |C|/|D|$
end while
```

4.2 Proposed method

We need to extend the subsumption algorithm so that the extracted hierarchy contains (almost) all objects (or documents). As the purpose of subsumption (Sanderson and Croft, 1999) is to derive concept hierarchy, it does not care how many of objects the terms occur. This can be controlled by the thresholds, but lower thresholds badly affect the suitability of the concept hierarchy. Therefore, we need another type of extension of subsumption algorithm to construct concept hierarchy covering (almost) all objects.

The basic idea of our extension of subsumption algorithm is to iteratively construct the concept hierarchy for uncovered objects until (almost) all objects are covered. We denote the whole set of objects as D and a set of covered objects in *i*-th iteration as Ci. At the beginning, we apply subsumption algorithm over D, and we obtain the concept hierarchy h1, then compute a set C1 of covered objects by h1. Then, for a set of uncovered objects U = D/C1, we apply subsumption algorithm to obtain a concept hierarchy h2. We continue these process until the ratio *rc* of covered objects UiCi among whole objects D becomes greater than threshold θ. The coverage ratio *rc* is computed as follows:

$$rc = \frac{|\cup_{i=1}^{k} C_i|}{|D|}$$

where k is the number of iterations, a.k.a. the number of constructed concept hierarchies. This procedure is summarized in Algorithm 1, where subsumption function returns the concept hierarchy for the object set U with given thresholds, and cover function returns a set of objects in D containing terms in h.

As a result, we obtain a set of hierarchies to navigate users through the whole set of objects. In the faceted search interface, we have other facets to restrict the set of objects. Any selection of a pair of facet and value changes the set of result objects. In addition, preparing such hierarchies for each set of result objects corresponding to the selections of pairs' facet and value is infeasible. Therefore, we need an online extension of the iterative subsumption algorithm. The essential requirement for online algorithm is to finish in a reasonable time. The iterative algorithm processes subsumption algorithm many times when the discoveries of hierarchies gradually increase the coverage. Hence, we stop the iteration in a limited number. The online algorithm is summarized in Algorithm. 2.

The interface shows the roots of the extracted hierarchies. For the current results, the hierarchies are computed, so, for further exploration, a user is navigated from the root to second level of the hierarchy. If there is only one hierarchy as a result of online iterative subsumption algorithm, we drill one level down the hierarchy and show the terms in the second level. Whenever a user selects/deselects any of facets and their values, the hierarchies are re-computed, except when the user selects textual facets, we can omit the re-computation of the hierarchy since the concept hierarchy is already computed.

Algorithm 2 Online iterative subsumption algorithm.
Input: a set of objects D, coverage threshold θ, iteration threshold δ, thresholds for subsumptions τ_s and τ_d
Output: A set of concept hierarchies H

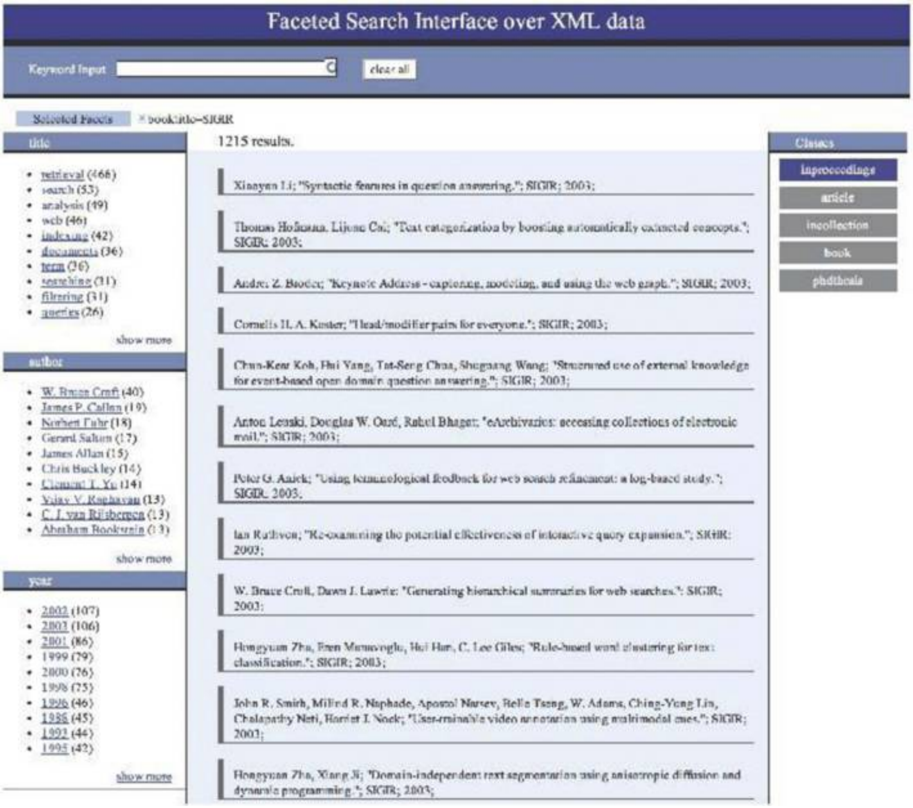
```
$H \leftarrow \{\}, C \leftarrow \{\}, U \leftarrow D$
$rc \leftarrow |C|/|D|$
$it \leftarrow 0$
while $rc < \theta$ and $it < \delta$ do
    $U \leftarrow D \backslash C$
    $h \leftarrow subsumption(U, \tau_s, \tau_d)$
    $C \leftarrow C \cup cover(D, h)$
    $H \leftarrow H \cup \{h\}$
    $rc \leftarrow |C|/|D|$
    $it \leftarrow it + 1$
end while
```

4.3 Faceted search interface

The snapshot of the faceted search interface over DBLP XML dataset is shown in Figure 2. On the interface, keyword search is available besides faceted search, so a user can execute keyword search from the topmost search box. The main panels consist of three components, namely, facet panel, result panel and class panel [as the class is defined in the previous work (Komamizu *et al.*, 2011), check it out for classes]. On the facet panel, facets and their values are shown with number of objects in the current results each value occurs. When a user clicks a value of a facet, she can restrict the current results to which containing the selected value of the facet. For instance, on Figure 2, when she clicks "2002" of year facet, she obtains result papers which are published in 2002. The selected values of facets are shown below the keyword search box (e.g. "SIGIR" on book title facet in Figure 2). On the result panel, result objects are shown. As our target data are XML data, the results should be XML, but for comfort on browsing, we convert result XML data as bibliographic information by XSLT. In the DBLP XML dataset, we choose title as a textual facet, so the values of the title facet are extracted terms instead of full title texts.

5. User study

We evaluate a facet-value extracted faceted search interface using the proposed approach, comparing with the conventional faceted search interface as explained in Komamizu *et al.* (2011) (introduced in Section 3). The expectation for this evaluation is that the extracted facet-values from textual contents help users navigate a set of result objects. To observe this, we perform a user study and observe how users can explore result objects through facets. As evaluating exploratory search on user study is dependent upon designed tasks, the design principle for tasks is one of the most important factors of user studying. Kules *et al.* (2009) suggest for designing faceted search tasks that each task should be ambiguous, discovery, in an unfamiliar domain and low-level description about what to find. However, the task design based on Kules *et al.* (2009) only focuses upon how the tasks should be formed. The other importance of



× booktitle=SIGIR

- retrieval (466)
- search (53)
- analysis (49)
- web (46)
- indexing (42)
- documents (36)
- term (36)
- searching (31)
- filtering (31)
- queries (26)

- W. Bruce Croft (40)
- James P. Callan (19)
- Norbert Fuhr (18)
- Gerard Salton (17)
- James Allan (15)
- Chris Buckley (14)
- Clement T. Yu (14)
- Vijay V. Raghavan (13)
- C. J. van Rijsbergen (13)
- Abraham Bookstein (13)

- 2002 (107)
- 2003 (106)
- 2001 (86)
- 1999 (79)
- 2000 (76)
- 1998 (75)
- 1996 (46)
- 1988 (45)
- 1993 (44)
- 1995 (42)

1215 results.

Xinyan Li; "Syntactic features in question answering."; SIGIR; 2003;

Thomas Hofmann, Lijuan Cai; "Text categorization by boosting automatically extracted concepts."; SIGIR; 2003;

Andrei Z. Broder; "Keynote Address - exploring, modeling, and using the web graph."; SIGIR; 2003;

Cornelis H. A. Koster; "Head/modifier pairs for everyone."; SIGIR; 2003;

Chun-Keat Koh, Hui Yang, Tat-Seng Chua, Shuguang Wang; "Structured use of external knowledge for event-based open domain question answering."; SIGIR; 2003;

Anton Leuski, Douglas W. Oard, Rahul Bhagat; "eArchivarius: accessing collections of electronic mail."; SIGIR; 2003;

Peter G. Anick; "Using terminological feedback for web search refinement: a log-based study."; SIGIR; 2003;

Ian Ruthven; "Re-examining the potential effectiveness of interactive query expansion."; SIGIR; 2003;

W. Bruce Croft, Dawn J. Lawrie; "Generating hierarchical summaries for web searches."; SIGIR; 2003;

Hongyuan Zha, Eren Manavoglu, Hui Han, C. Lee Giles; "Rule-based word clustering for text classification."; SIGIR; 2003;

John R. Smith, Milind R. Naphade, Apostol Natsev, Belle Tseng, W. Adams, Ching-Yung Lin, Chalapathy Neti, Harriet J. Nock; "User-trainable video annotation using multimodal cues."; SIGIR; 2003;

Hongyuan Zha, Xiang Ji; "Domain-independent text segmentation using anisotropic diffusion and dynamic programming."; SIGIR; 2003;

Figure 2.
A snapshot of the prototypical interface over DBLP XML dataset. The value "SIGIR" of the book title facet is selected. On the left side, three facets are shown, namely, title, author and year, and the title is textual facet. In the middle, the results of papers are shown

user study is specificity of tasks. We are going to discuss how our evaluation tasks are designed in Section 5.2. Before it, we briefly explain about experimental settings in Section 5.1. Section 5.3 examines the sufficiency of subsumption algorithm to extract topical terms as a preliminary experiment. Then, we describe experimental methodology using the designed tasks in Section 5.4, and we show the experimental results in Section 5.5.

5.1 Settings

In this experiment, we compare the following three methods:

(1) MLCA-based keyword search (Li *et al.*, 2004);

(2) faceted search with keyword search (Komamizu *et al.*, 2011); and

(3) the proposed approach with keyword search.

Due to the fact that only exact matching of keyword search degrades its search performance, we utilize stemming (Porter, 1980) techniques. The reason why every method is combined with keyword search is that its familiarity with faceted search, and we have shown the good effect of combining keyword search with faceted search in Komamizu *et al.* (2011).

The dataset used in this experiment is DBLP XML dataset (Team, 2006), which is the XML data containing bibliographic information about computer science researches. The dataset consists of several kinds of paper elements (e.g. in proceedings and books) under the root element named DBLP. Using the framework, we can obtain these kinds of papers as objects and its descending elements as facets (the selected classes and facets are shown in Table II). In this dataset, the title facet is detected as a textual facet by Definition 3 and we extract facet-values from textual contents of title elements in the XML data.

5.2 Task design

As is mentioned in Kules *et al.* (2009) 13, tasks should be understandable and possible to imagine the situation that the users stand for. So, each task is a scenario to find one or more objects in the dataset. A template of task scenario is as follows (this is copied from Kules *et al.*, 2009):

> Imagine that you are taking a class called. For this class, you need to write a paper on the topic. Use the catalog to find two possible topics for your paper. Find three books for each topic.

We arrange this template to design tasks. The main difficulty of designing tasks on this template is to determine the topic terms in the second blanked line. We call such terms as task terms. For stable evaluation, criteria to choose the task terms are required.

In this paper, we give a criterion to choose task terms from the given dataset. Basic idea of the criterion is based on the occurrence of the task terms among the dataset. Terms occurring most of objects in the dataset are regarded as general terms at least in the domain of the dataset ("approach" for example in DBLP XML dataset). While, terms occurring fewer objects are regarded as specific terms in the dataset, e.g. special name of application. When the given task terms are general, users are required to explore large number of objects to find the direction to achieve the task. Hence, exploratory search tasks should include relatively general task terms, and the exploratory search methods should help users explore the directions easily. On contrary, the given task terms are specific, users can find desired objects when they use *ad hoc* search methods. Thus, there exists the trade-off between specificity of task terms and the performance of exploratory search methods against *ad hoc* search methods. To represent this trade-off, the criterion proposed in this paper is called specification level which indicates how each task is specific on a limited number of objects. The specification level sl(T) of a task T is defined as conjunct selectivity of terms in the task (Definition 4).

Definition 4 (specification level): Given a task $T = \{t1, t2, \ldots, t|T|\}$ which consists of task terms ti, the specification level sl(T) of T is defined as conjunct selectivity of terms as follows:

$$sl(T) = \frac{|\cap_{ti\in T}\,\psi_{ti}(D)|}{|D|}$$

where D is the whole objects and ψt_i (D) is a selection function which returns matching objects with the input keyword ti.

Note that the smaller specification level a task has, the more specific the task is. The specification level can be also understood as document frequency of task terms. As specification level is document frequency, specification level monotonically decreases when add a term to the current terms (Lemma 1).

Lemma 1 (monotonicity of specification level): Given a task T and any term t ϵ/T, the specification levels of T and $T' = T \cup \{t\}$ hold the following condition:

$$\mathrm{sl(T)} \geq \mathrm{sl(T')}$$

□

Proof 1 (monotonicity of specification level): Given a task T and any term u/ϵ T, the specification level of T is calculated as follows:

$$sl(T) = \frac{|\cap_{ti\in T}\,\psi_{ti}(D)|}{|D|}$$

The specification level of the task $T' = T \cup \{u\}$ which u is added to T is calculated as follows:

$$sl(T') = \frac{|\cap_{ti\in T}\,\psi_{ti}(D)|}{|D|} = \frac{|\psi_u(D) \cap \cap_{ti\in T}\,\psi_{t_i}(D)|}{|D|}$$

The size of the intersection of two sets is smaller or equals to the size of the smallest one among the two sets. Thus, the numerators of these specification levels hold the following

Class	facet
Article	Editor, title, journal, year, author and publisher
Phd thesis	Author, title, year, series and publisher
In proceedings	Author, title, year and book title
Book	Author, title, publisher, year, editor, book title and series
In collection	Author, title, year, book title and publisher

Table II. Selected classes and facets in DBLP XML dataset. The underlined facets are applied our approach to extract values

condition: $|\cap_{t_i \in T} \psi_{t_i}(D)| \geq |\cap_{t_i \in T'} \psi_{t_i}(D)|$. Hence, as the denominators of these specification share the same value, the specification levels of them hold the following condition: $\mathrm{sl(T)} \geq \mathrm{sl(T')}$. ■

Using specification level, we can automatically generate search tasks for given specification level by automatically calculating task terms. For choosing one term as the task term, it is straightforward that choosing a term which specification level is close to the given specification level. For choosing two terms as the task terms, according to Lemma 1, firstly, we choose one term which specification level is greater than the given specification level, and then choose another term which combination with the previously chosen term satisfy the given specification level. Similarly, we can compute tasks which number of task terms is more than two. At last, the examiner should validate whether chosen task terms are appropriate and modify the scenario sentences. Table III demonstrates specification levels for single and multiple task terms extracted from title elements in the DBLP XML dataset.

An example of tasks in this evaluation is like following:

> Imagine that you are taking a class called Introduction to Machine Learning. For this class, you need to write a paper on the topic support vector machines. Use the database to find two possible topics for your paper. Find three papers for each topic.

The terms "support vector machines" (specification level is 0.00052) are taken from extracted task terms in Table III, and we put "Introduction to Machine Learning" as the name of the class. To put the name of class, the examiner should be careful not to put more specific terms than task terms to the name, as the class name should not decrease the specification level of the task.

5.3 Preliminary experiment

Before evaluating the proposed scheme, we conduct the preliminary experiments to evaluate feasibility of the subsumption algorithm to our setting. We evaluate whether the proposed method using subsumption algorithm extracts reasonable terms from textual contents.

For this evaluation, we compare the proposed scheme with various approaches of term selection listed as follows: random selection, entropy maximization, coverage maximization and frequency-based selection. The random selection approach selects terms from whole set of terms contained in the textual contents at uniformly random. The entropy maximization approach chooses a set of terms which maximize entropy of the selected terms. As finding terms with entropy maximization can be reduced to set cover problem (Korte and Vygen, 2012) which is an NP-hard, we apply a greedy algorithm. The coverage maximization approach extracts a set of terms iteratively. For each iteration, a residual set of objects is target and extract term which covers maximally the residual set of objects. The frequency-based selection approach selects a set of the most frequent terms.

This preliminary experiment is also user study that we ask the users feasibility of suggested terms in the context of restricted conditions. The restricted conditions are situated by task terms with regard to specification levels. We show all extracted terms by these approaches in a mixed set, and the users choose some of them which are feasible for each situation. For instance, in the situation "support vector machine", the users are

Term	Specification level
Analysis	0.03534
Design	0.03198
Database	0.01713
Graph	0.00755
Large	0.00746
Security	0.00549
Neural networks	0.00484
Case study	0.00482
Logic programming	0.00366
User interface	0.00173
Knowledge representation	0.00148
Relational database	0.00115
World Wide Web	0.00110
Support, vector machines	0.00052
Inductive logic programming	0.00035
Analysis case study	0.00026

Table III. Specification levels of sampled terms in the DBLP dataset

shown “kernel”, “linear”, “application” and so on, and then they choose feasible terms for further restrictions for their explorations.

For evaluation, we calculate precision of extracted terms in each approach. The precision is calculated as a ratio of ground truth terms within the extracted terms. Figure 3 shows the result of the preliminary experiment. Each bar in the figure represents precision score of the corresponding approach. Horizontal axis shows specification levels, and vertical axis shows precision scores.

From the figure, the proposed scheme using subsumption algorithm perform the best among the approaches. The difference from the random approach indicates a significance of the proposed approach. Also, the proposed approach is constantly better than the frequency-based approach except the case which the specification level is 0.002. That means, the proposed approach is not too straightforward approach to extract terms from the textual facets.

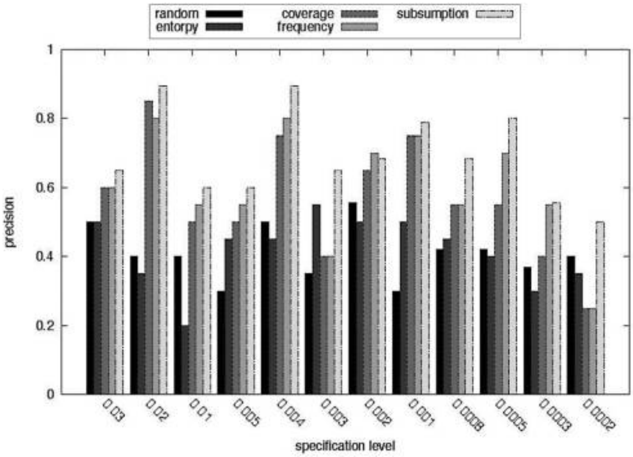
5.4 Methodology

To evaluate the proposed approach, we measure its usability comparing with the conventional faceted search introduced in Komamizu *et al.* (2011) and the conventional *ad hoc* search method for XML data, i.e. keyword search. To evaluate in terms of usability, we measure time that the users consume to achieve tasks, the number of clicks on facets and the number of keyword search performed, by varying specification levels of tasks.

In this evaluation, we have five male and female volunteer users whose age is between 22 and 30. They have research experiences in computer science fields. We perform the same tasks for all the users, and we observe the average measurement over the users.

5.5 Results

In this section, we describe the results of our experimental evaluations. The experimental results for time consumed to achieve the given tasks are summarized



Note: Random selection, entropy maximization, coverage maximization, frequency-based selection and subsumption

Figure 3.
Precision of facet-value extraction approaches using five approaches

in Figure 4, and those of the number of operations performed are summarized in Figures 5-7. In the rest of this section, we analyse these results in detail.

The results in Figure 4 indicate that our proposed approach achieve the best performance among the tasks which have various specification levels. Comparisons between keyword search method and faceted search with keyword search show that our precious work (Komamizu *et al.*, 2011) outperforms the MLCA-based keyword search. The previous work improves about 21 per cent on average and about 47.1 per cent maximum. This is because, the facets support users to find objects, while users in keyword search must find appropriate keywords from the current result objects by themselves. Moreover, the proposed approach outperforms both of them. Our proposed approach improves the keyword search method about 56.1 per cent on average and 77.8 per cent maximum, and improves the previous approach about 44.4 per cent on average and about 68.9 per cent maximum. This result indicates that extracting appropriate terms from textual facets helps users overview the current results and successfully restrict by the terms to achieve the tasks.

Figures 5-7 show how the users search using the methods in terms of operations, which are keyword search and facet selection. Figure 5 represents average number of operations until achieving the tasks corresponding with specification levels. From this figure, we observe that even though keyword search method takes more time than others (Figure 4), the numbers of operations performed are not significantly high. Thus, we need to observe in more detail.

Figures 6 and 7 show the number of each operations, namely, keyword search operations and facet selections, respectively.

Figure 6 depicts the average number of keyword search operations performed until users finish the tasks. As is expected, our previous work decreases the number of keyword search performed with some exceptions. This indicates that the previous work can navigate users when the facets appropriately categorize the current result objects. The succeeded case of the previous work is that, when users look for keywords to

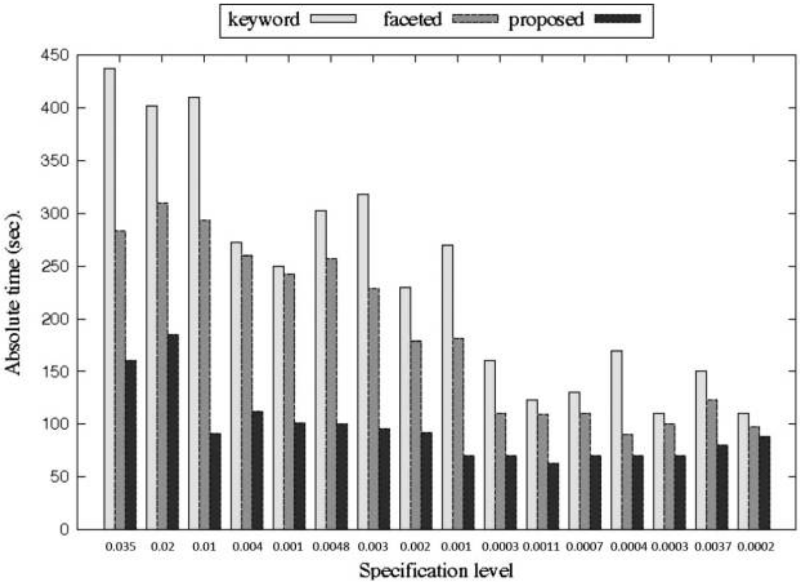


Figure 4.
Average time (sec) consumed to achieve the tasks

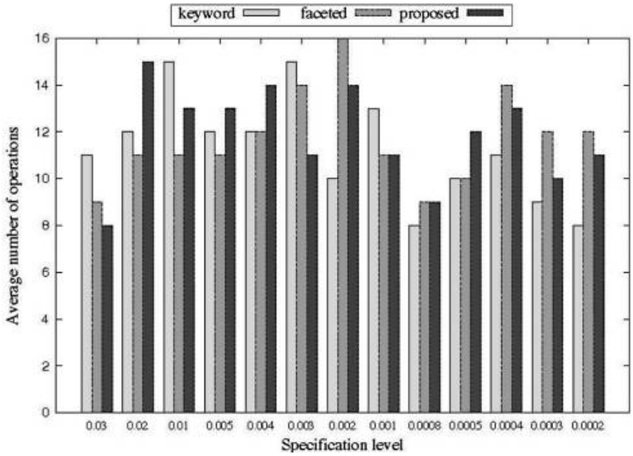


Figure 5.
Total number of operations

restrict the current results, they use year facet to restrict the result objects. Then, the users find a good keyword to restrict and performs keyword search and then obtain the desired results. On our proposed approach, the numbers of keyword search are almost constant in low number. In our proposed approach, a user requires keyword search to restrict the objects to the limited number of objects matching with the task terms. Thus, this figure indicates that once the users search by task terms, they use only facet selections to restrict the result objects.

Figure 7 shows the number of facets used for finding desired result objects. Obviously, as keyword search does not have any option to use facets, the numbers

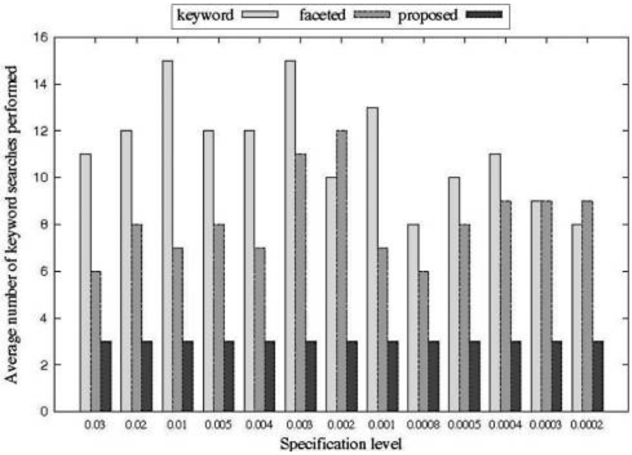


Figure 6.
The number of keyword search performed

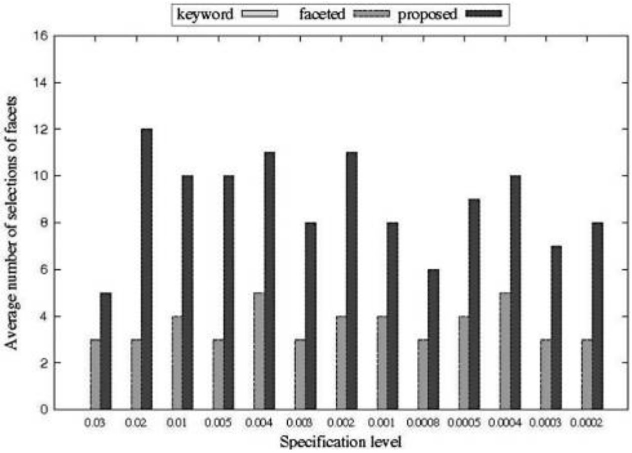


Figure 7.
The number of facets selected

for keyword search are all zero. The numbers of facets used in our proposed approach are much larger than that of faceted search with keyword search. This means that our proposed approach gives a lot of informative values of facets to the users. Considering with the result of the number of keyword search shown in Figure 6, the values of facets extract in our approach are selected to restrict the current results. The average usage of values in textual facets over all facet selections in our proposed approach is 95.3 per cent, that is once users perform keyword search on task terms, they select values of textual facets in most of cases. Futhermore, analyzing with the result for elapsed time shown in Figure 4, the values of facets in our approach have nicer overview of the current results and suggest values of facets for further restrictions.

As a result, even though the number of selected facets is large on our proposed method, the time consumed to achieve task is smaller than others. This is because, among all methods, users need to see the current results and choose next actions. On keyword search, users need to carefully see the current results to find the next input keyword; however, this requires high efforts. On the other hand, seeing facets and their values, which give an overview of the current results, requires less effort. In addition, keyword search implies human error sometimes like typos, while users just click on values of facets in faceted search which do not imply such errors.

Table IV shows the correlations between specification level and experimental measurements, namely, the average time, the average number of operations, the average number of keyword search performed and the average number of facet selections. The correlation between specification and the average time is high for every methods. This can be observed from Figure 4 as well. The figure shows that users consume more time for more specific tasks, and this phenomenon is almost common in these methods. This is what we expect on the specificity of tasks, that is users take more time for tasks with high specification levels (i.e. vague tasks). This implies that we can control the tasks by specification levels for the consumption time of each task. On the other hands, the number of operations performed in each task is low correlation with specification levels (all in range [−0.5:0.5]). This is an interesting observation, because the number of operations is considered to be one of robust experimental measurements for efficiency of exploratory search.

6. Conclusion and future work

In this paper, we propose an extraction mechanism of values of facets which have longer and unique texts for corresponding objects. Also, we propose an evaluation methodology using specification levels. Then, we analyse the stability of evaluation using specification levels. Experimental evaluation of our proposed approach using the proposed evaluation methodology shows that our proposed approach improves search performance comparing with the previous approach Komamizu *et al.* (2011) and keyword search approach.

For the future work, we will extend our proposed approaches for various situations, such as faceted search for heterogeneous XML data, faceted search for graph structured data and faceted search for more specific domain like chemistry. Also, we are going to explore evaluation methodologies which achieve high reproductivity.

Measurement	vs Keyword	vs Faceted	vs Proposed
Time	0.771346	0.655485	0.841967
# Operations	0.236971	−0.411838	−0.122954
# Keywords	0.236971	−0.39138	−0.063875
# Facets	–	−0.299988	−0.122954

Notes: The average time (referred as time), the average number of operations (referred as # operations), the average number of keyword search performed (referred as # keywords), and the average number of facet selections (referred as # facets)

Table IV. Correlations between specification level and measurements

Notes

1. www.w3.org/TR/REC-rdf-syntax/
2. www.w3.org/TR/soap/
3. web.expasy.org/docs/swiss-prot_guideline.html
4. www.genome.jp/kegg/
5. www.ebxml.org/
6. www.xbrl.org/
7. www.informatik.uni-trier.de/∼ley/db/
8. www.ebay.com/

References

Abel, F., Celik, I., Houben, G.-J. and Siehndel, P. (2011), "Leveraging the semantics of tweets for adaptive faceted search on Twitter", *International Semantic Web Conference*, Vol. 1, pp. 1-17.

Dakka, W., Ipeirotis, P.G. and Wood, K.R. (2005), "Automatic construction of multifaceted browsing interfaces", *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, *ACM, New York, NY*, pp. 768-775.

Dash, D., Rao, J., Megiddo, N., Ailamaki, A. and Lohman, G.M. (2008), "Dynamic faceted search for discovery-driven analysis", *CIKM*, ACM, New York, NY, pp. 3-12.

Goldman, R. and Widom, J. (1997), "DataGuides: enabling query formulation and optimization in semistructured databases", *International Conference on Very Large Data Bases*, pp. 436-445.

Hahn, R., Bizer, C., Sahnwaldt, C., Herta, C., Robinson, S., Bürgle, M., Düwiger, H. and Scheel, U. (2010), "Faceted Wikipedia search", *13th International Conference on Business Information Systems*, pp. 1-11.

Kashyap, A., Hristidis, V. and Petropoulos, M. (2010), "FACeTOR: cost-driven exploration of faceted query results", *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, *ACM, New York, NY*, pp. 719-728.

Komamizu, T., Amagasa, T. and Kitagawa, H. (2011), "A framework of faceted navigation for XML data", *Proceedings of the 13th International Conference on Information Integration and Web-based Applications and Services*, *ACM, New York, NY*, pp. 28-35.

Komamizu, T., Amagasa, T. and Kitagawa, H. (2014), "A scheme of automated object and facet extraction for faceted search over XML data", *Proceedings of the 18th International Database Engineering & Applications Symposium*, *ACM, New York, NY*, pp. 338-341.

Kong, W. and Allan, J. (2013), "Extracting query facets from search results", *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, *ACM, New York, NY*, pp. 93-102.

Korte, B. and Vygen, J. (2012), *Combinatorial Optimization: Theory and Algorithms (Algorithms and Combinatorics)*, Springer-Verlag Berlin Heidelberg.

Koren, J., Leung, A., Zhang, Y., Maltzahn, C., Ames, S. and Miller, E.L. (2007), "Searching and navigating petabyte-scale file systems based on facets", *Proceedings of the 2nd International Workshop on Petascale Data Storage: Held in Conjunction with Supercomputing*, *ACM, New York, NY*, pp. 21-25.

Koren, J., Zhang, Y. and Liu, X. (2008), "Personalized interactive faceted search", *Proceedings of the 17th International Conference on World Wide Web*, *ACM, New York, NY*, pp. 477-486.

Kules, B., Capra, R., Banta, M. and Sierra, T. (2009), “What do exploratory searchers look at in a faceted search interface?”, *Proceedings of the 9th ACM/IEEE-CS Joint Conference on Digital Libraries, ACM, New York, NY*, pp. 313-322.

Li, C., Yan, N., Roy, S.B., Lisham, L. and Das, G. (2010), “Facetedpedia: dynamic generation of query-dependent faceted interfaces for Wikipedia”, *Proceedings of the 19th International Conference on World Wide Web, ACM, New York, NY*, pp. 651-660.

Li, G., Feng, J., Wang, J. and Zhou, L. (2007), “Effective keyword search for valuable LCAs over XML documents”, *CIKM Proceedings of the 16th ACM Conference on Information and Knowledge Management, ACM, New York, NY*, pp. 31-40.

Li, Y., Yu, C. and Jagadish, H.V. (2004), “Schema-free XQuery”, *Proceedings of the 13th International Conference on Very Large Data Bases*, pp. 72-83.

Marwick, A. (2008), “Faceted navigation for document discovery”, available at: www.ibm.com/developerworks/data/library/techarticle/dm-0802marwick/

Oren, E., Delbru, R. and Decker, S. (2006), “Extending faceted navigation for RDF data”, *International Semantic Web Conference*, Athens, GA, pp. 559-572.

Porter, M. (1980), “An algorithm for suffix stripping”, *Program*, Vol. 14 No. 3, pp. 130-137.

Pound, J., Paparizos, S. and Tsaparas, P. (2011), “Facet discovery for structured web search: a query-log mining approach”, *SIGMOD Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data, ACM, New York, NY*, pp. 169-180.

Roy, S.B., Wang, H., Das, G., Nambiar, U. and Mohania, M.K. (2008), “Minimum-effort driven dynamic faceted search in structured databases”, *Proceeding of CIKM, ACM, New York, NY*, pp. 13-22.

Sanderson, M. and Croft, W.B. (1999), “Deriving concept hierarchies from text”, *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 206-213.

Stoica, E., Hearst, M.A. and Richardson, M. (2007), “Automating creation of hierarchical faceted metadata structures”, *HLT-NAACL Proceedings of the Human Language Technology Conference*, pp. 244-251.

Team, T.D. (2006), “The DBLP computer science bibliography”, available at: www.informatik.uni-trier.de/~ley/db/

Tunkelang, D. (2009), *Faceted Search, Synthesis Lectures on Information Concepts, Retrieval, and Services*, Morgan & Claypool Publishers.

Wang, J., Peng, X., Xing, Z. and Zhao, W. (2013), “Improving feature location practice with multi-faceted interactive exploration”, *35th International Conference on Software Engineering*, San Francisco, CA, pp. 762-771.

White, R.W., Kules, B., Drucker, S.M. and Schraefel, M. (2006), “Supporting exploratory search”, *Communication of ACM*, Vol. 49 No. 4.

W.W.W.C. (W3C) (2015), “Extensible markup language (XML)”, available at: www.w3.org/XML/

Xu, Y. and Papakonstantinou, Y. (2005), “Efficient keyword search for smallest LCAs in XML databases”, *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data, ACM, New York, NY*, pp. 537-538.

Yee, K.-P., Swearingen, K., Li, K. and Hearst, M.A. (2003), “Faceted metadata for image search and browsing”, *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ACM, New York, NY*, pp. 401-408.

About the authors

Takahiro Komamizu received BE, ME and PhD in Engineering from University of Tsukuba in 2009, 2011 and 2015, respectively. Currently, he is a Research Fellow at Center for Computational Sciences, University of Tsukuba, Japan. His research interests include database systems, information retrieval, usability study, XML data management, data mining and multimedia data management. He is a member of ACM and DBSJ. Takahiro Komamizu is the corresponding author and can be contacted at: taka-coma@acm.org

Toshiyuki Amagasa is an Associate Professor at Faculty of Engineering, Information and Systems, University of Tsukuba. His research interests include data engineering, Web information management, scientific information management and data mining. He is a senior member of IEICE and IEEE, and a member of DBSJ, IPSJ and ACM.

Hiroyuki Kitagawa received the BSc degree in physics and the MSc and DrSc degrees in computer science, all from the University of Tokyo, in 1978, 1980 and 1987, respectively. He is currently a Professor at Faculty of Engineering, Information and Systems and at Center for Computational Sciences, University of Tsukuba. He serves as President of the Database Society of Japan. His research interests include data integration, databases, data mining and information retrieval. He is an IEICE Fellow, an IPSJ Fellow, a member of ACM and IEEE and an Associate Member of the Science Council of Japan.