High-speed Detection of Ontological Knowledge and Bi-directional Lexico-Syntactic Patterns from the Web

Hiroaki Ohshima and Katsumi Tanaka
Department of Social Informatics, Graduate School of Informatics, Kyoto University, Japan
Email: {ohshima, tanaka}@dl.kuis.kyoto-u.ac.jp

***Abstract*— We propose a high-speed method of detecting ontological knowledge from the Web. Ontological knowledge in this paper means a term related to a given term. For example, hypernyms and hyponyms are basic related terms that are treated in dictionaries. Synonyms and coordinate terms are also well-defined related terms. Topic terms and description terms represent topics of the given term and they are vaguely defined. There are other related terms such as abbreviations and nicknames. The proposed method can be used for detecting many kinds of related terms. It extracts related terms from text resources only from Web search results, which consist of the titles, snippets, and URLs of Web pages. We use two different kinds of lexico-syntactic patterns to extract related terms from the search results, and these are called bi-directional lexico-syntactic patterns. The proposed method can be applied to both languages where words are separated by a space such as English and Korean and ones where words are not separated by a space such as Japanese and Chinese. The proposed method does not need any advanced natural language processing such as morphological analysis or syntactic parsing. It works relatively fast and has excellent precision. We also propose a method of automatically discovering superior bi-directional lexico-syntactic patterns using Web search engines because it is sometimes difficult to find appropriate patterns to detect related terms in a certain relationship.**

***Index Terms*— Knowledge search, Knowledge acquisition**

I. Introduction

Dictionaries are used in many services and applications, and the knowledge in these improves the effectiveness or reliability of the services and applications. Content in dictionaries is usually static. However, if necessary knowledge could be dynamically obtained in real time, it would be acceptable to use such dynamic dictionaries in services and applications. Suppose that applications allowed users to issue keyword queries. It would then be very difficult to assume the range of queries that users would issue. When the applications wanted to use dictionaries to obtain terms related to users' given queries, the queries would need to be contained in the dictionaries as entry words. Because no dictionary can cover all terms in all fields, we need to dynamically detect terms related to a given term.

The Web has been used as a huge data resource due to its current growth. Everybody can easily access necessary information through the great number of search interfaces that are available for it. We believe that knowledge can be rapidly detected with excellent precision using Web search engines.

This paper proposes a high-speed method of detecting terms in a certain relationship using Web search engines. There are many kinds of "related terms". Some are well-defined, such as hypernyms, hyponyms, synonyms, and coordinate terms[1]. Some are vaguely defined, such as topic and description terms. Even the abbreviations, nicknames, and attributes of an object can be regarded as related terms.

Knowledge is extracted from many kinds of resources. Traditional research has used huge text corpora. However, the Web is currently often used for research. There are several methods of using Web data: by gathering huge numbers of Web pages by using crawlers, by downloading Web pages found in Web search results, and by only using titles and snippets of Web pages in Web search results. The amount of text used in a particular method affects the speed, precision, and completeness of the results. When more data are used, generally speed slows, precision increases, and more related terms are obtained.

Our previous work [1] focused on discovering the coordinate terms of a given term. It only used Web search results (URLs, titles, and snippets) as data resources without downloading original Web pages. In that method, we focused on the fact that coordinate terms are often connected with the conjunction "*or*". When a term was given, we collected Web search results where a query was a text string connecting "*or*" and the given term. The coordinate terms of the given term were only extracted from the text in the search results. The method was quite fast and had excellent precision. For example, when a total of 200 search results was used, it took 3.3 sec, more than 12 coordinate terms were returned, and the precision of the method was over 80% [2]. Consequently, it was deemed suitable for use with many kinds of services and applications.

In other previous work [2], we proposed a generalized method of detecting coordinate terms, which achieved high-speed detection not only for coordinate terms but also for many kinds of related terms. This paper proposes an extension to this generalized method that is more flexible with numerous kinds of relationships.

[1]Coordinate terms are terms that have the same hypernym.

doi:10.4304/jsw.5.2.195-205

As it is occasionally difficult to find appropriate lexico-syntactic patterns for detecting related terms in a certain relationship, we also propose a method of automatically discovering superior patterns. When a pair of terms is given, the method find patterns that detect related terms in the relationship between the given terms using Web search engines.

The rest of the paper is organized as follows. Section II discusses related work, and Section III describes details on the high-speed detection of related terms using lexico-syntactic patterns. Section IV describes details on a method of automatically discovering useful lexico-syntactic patterns when a sample term pair is given. We conclude the paper in Section V.

II. Related Work

When an application needs to use knowledge about relationships between terms, the easiest way is to use electronic dictionaries that are either on the Web or not . One of the most popular dictionaries is WordNet[2] [3], a lexical database for English. As it contains a hierarchical thesaurus, we can obtain the hyponyms and hypernyms of any term. It is also possible to obtain the coordinate entries of a term by finding terms that share the same hypernym. There are also various electronic dictionaries on the Web, such as Wikipedia[3] and Wiktionary[4]. Although they provide accurate knowledge, no dictionary can cover all terms in all fields, and they do not support all kinds of relationships between terms. Although these dictionaries are quite useful, it is still necessary to detect many kinds of related terms on demand.

There are many methods of extracting knowledge from huge text corpora. Hearst [4] proposed methods of extracting hypernyms and hyponyms. She used several lexico-syntactic patterns such as "<hypernym> *such as* <hyponym>". The Lexico-syntactic patterns that Hearst developed have been used by many other researchers. Etzioni et al. [5] proposed a system called KnowItAll, where knowledge is extracted using Hearst's patterns and various other patterns. Each of the lexico-syntactic patterns is independently used to detect related terms. For example, they used morphological analysis and obtained noun phrases to extract correct words.

Ghahramani et al. [6] proposed a method called Bayesian Sets to acquire coordinate terms. It found clusters of terms based on Bayesian inference. Although their algorithm was simple and fast, it needed a large data set such as the Grolier encyclopedia or EachMovie.

Lin [7] proposed a method of making clusters of similar words. Similarities between words were calculated and clusters containing similar words were generated. Modification relation was used to calculate the similarities. Therefore, a large corpus with modification relation was necessary for his method.

Shinzato and Torizawa [8] proposed a method of acquiring coordinate terms from HTML documents. They focused attention on HTML structures. Terms in the structures on the same level such as itemized terms in a list could become candidates for coordinate terms. If mutual information and cooccurrence of terms were high, they were regarded as coordinate terms. The same researchers [9] also proposed a method of discovering hypernyms and hyponyms using HTML structures.

Google Sets[5] is an interesting tool for acquiring terms in the same class. Consider a set that consists of coordinate terms. When we enter a small subset of the items, Google Sets returns the whole set. For example, when a query for Google Sets consists of *Versace* and *Armani*, the result shows a list of *Versace*, *Armani*, *Gucci*, *Chanel*, *Prada*, *Calvin Klein*, and other fashion names. Details on the Google Sets algorithm have not been disclosed, but it seems that a large-scale clustering algorithm is applied to collected Web pages where many clusters of terms have already been generated. This can be described as the automatic construction of a coordinate-term dictionary. If that is the case, Google Sets operates similarly to other work that uses huge amounts of data.

Yamaguchi et al. [10] proposed a method of discovering the related terms of a given term from the query log data of a Web search engine. The coordinate terms and topic terms were extracted. The topic terms of a given term indicate typical terms that describe the given term, and they may contain hypernyms, hyponyms, and attribute terms. There have been several other projects to extract ontological knowledge from query logs [11]–[14].

There are many other researchers who have discovered many kinds of related terms. Sanderson and Croft [15] proposed a method of extracting a concept hierarchy based on the subset relationships between sets of documents. They attempted to find particular expressions that indicated meaningful relationships in many documents.

Glover et al. [16] proposed a method of determining parent, self, and child keywords for a set of Web pages, where self words described the cluster itself and parent and child words corresponded to descriptions of more general and more specific concepts.

Oyama and Tanaka [17] proposed a method of identifying pairs of keywords in which one word described the other. They used hit counts from a Web search engine to measure relatedness, i.e., how a term described the other term. They focused on where terms appeared in a document, i.e., in the title or in the body. The operators "intitle:" and "intext:" on a Web search engine were used for this purpose.

Nakayama et al. [18], [19] proposed a method of generating a huge association thesaurus based on link mining from Wikipedia entities. The density of links between two Wikipedia entities was used to calculate their relatedness. Users could obtain several terms associated with a given term from the generated thesaurus. The

[2]http://wordnet.princeton.edu/
[3]http://wikipedia.org/
[4]http://wiktionary.org/

[5]http://labs.google.com/sets

thesaurus is currently available on the Web[6].

Hokama and Kitagawa [20] proposed a method of detecting the nicknames and mnemonic names of a person. Web search engines were used to collect pages that contained a person's name and her/his nickname. A lexico-syntactic pattern "<mnemonic name> *Ko To* <person's full name>" in Japanese was used to extract nicknames and mnemonic names from the pages.[7] Their assumption was that a person's name and her/his nickname often occurred in similar contexts. This means an appropriate nickname only occurs very often when the person's name appears.

There have been several researchers who have calculated the relatedness between two terms. Church and Hanks [21] proposed a method of calculating the semantic relationships between terms using mutual information. Their idea was that mutual information in semantically related terms was high, and this has often been used in other work to find similar terms such as hypernyms, hyponyms, and coordinate terms.

Turney [22] and Baroni and Bisi [23] proposed methods of discovering synonyms where cooccurrence or mutual information of target terms was calculated using hit counts from a Web search engine. This indicated that data in a Web search engine could be an alternative to data obtained from analyzing conventional large corpora. Google Similarity Distance [24] is also a method used to calculate similarities between terms.

Methods of calculating the relatedness between pairs of terms in many kinds of relationships have been proposed. Turney and Littman [25], [26] and Bollegala et al. [27] proposed methods of calculating the relatedness between two words in any kind of relationship. The target relationship could be changed. When a pair of terms was given, the relationship in the pair became the target relationship. The methods could then allow users to calculate the relatedness between any other pair of terms.

III. High-speed Detection of Related Terms

This section discusses the high-speed detection of related terms using bi-directional lexico-syntactic patterns from Web search results. First, we explain a formulation of the method in typical cases. Then, we present concrete examples that are used to detect for hypernyms or coordinate terms. After the features of the method are clarified, we discuss our extension of the method to more general cases.

A. Formulation

The proposed method returns related terms when a term is given. It uses two different kinds of lexico-syntactic patterns to extract related terms from text resources. The text resources are gathered from Web search results.

What we need to do first with the method is to determine two different kinds of lexico-syntactic patterns. The patterns are generally represented as

$$\begin{aligned} \mathbf{Pattern}^{Pre} &:= \text{prefix} \parallel \texttt{<t>} \\ \mathbf{Pattern}^{Post} &:= \texttt{<t>} \parallel \text{suffix} \end{aligned}$$

where $\parallel$ represents the concatenation of text strings, and <t> denotes one of terms related to the given term.

For example, when we want to detect the hypernyms of a given term, the following lexico-syntactic patterns can be used.

$$\begin{aligned} \mathbf{Patterns}^{Pre} &:= \texttt{<s>} \parallel \textit{ are} \parallel \texttt{<t>} \\ \mathbf{Patterns}^{Post} &:= \texttt{<t>} \parallel \textit{ such as} \parallel \texttt{<s>} \end{aligned}$$

where <s> denotes the given term.

When a certain pair of terms is related in a particular relationship, we assume that phrases exist that contain these terms. <s> and <t> can appear in two different kinds of phrases as follows.

- <s> ... <t>
- <t> ... <s>

<s> appears *before* <t> in some phrases, and <s> appears *after* <t> in some phrases.

For example, when we try to detect the hypernyms of a given term, we think about phrases that contain <s> and <t> as follows.

- <s> *are* <t>
- <t> *such as* <s>

Here, "<s> *are*" is a prefix of <t>[8], and "*such as* <s>" is a suffix of <t>. They are different kinds of lexico-syntactic patterns for detecting <t>. Not only are the patterns themselves different, but the order of <s> and <t> is also different.

We use two such different kinds of lexico-syntactic patterns to extract related terms from the search results, and they are called **bi-directional lexico-syntactic patterns**.

Web search results are gathered according to the patterns. The queries for Web search engines can be automatically created as follows.

$$Q^{Pre} := \begin{cases} \text{prefix} & (\text{if the prefix contains } \texttt{<s>}) \\ \text{prefix} \wedge \texttt{<s>} & (\text{if the prefix doesn't contain } \texttt{<s>}) \end{cases}$$

$$Q^{Post} := \begin{cases} \text{suffix} & (\text{if the suffix contains } \texttt{<s>}) \\ \text{suffix} \wedge \texttt{<s>} & (\text{if the suffix doesn't contain } \texttt{<s>}) \end{cases}$$

If the prefix and the suffix contain the given term <s>, they can be queries for Web search engines.

The previous work [2] only treated such cases. However, the prefix and the suffix do not have to contain <s>, and we add <s> to the queries for Web search engines in such cases.

[6] http://wikipedia-lab.org:8080/WikipediaThesaurusV2/

[7] Every underlined portion originally represents one character in Japanese. "*Ko To*" in Japanese almost means "*alias*".

[8] We should take into account the differences between articles; *a*, *an*, and *the*, and the absence of articles. This problem can easily be solved because a regular expression can be used to represent a pattern in practical implementations.

Next, they are issued to a Web search engine, and Web search results are gathered. The result obtained with the proposed method is a set of related terms that are extracted from the Web search results. This is denoted by T, and is represented as

$$\begin{aligned} T = \{\texttt{<t>} \mid & \quad \mathbf{Patterns}^{Pre} \in Parts(\quad Q^{Pre}) \\ & \wedge \quad \mathbf{Patterns}^{Post} \in Parts(\quad Q^{Post})\} \\ = \{\texttt{<t>} \mid & \quad (\text{prefix} \parallel \texttt{<t>}) \in Parts(\quad Q^{Pre}) \\ & \wedge \quad (\texttt{<t>} \parallel \text{suffix}) \in Parts(\quad Q^{Post})\} \end{aligned}$$

where $Parts(\quad Q)$ is a multiset of any phrase in the titles and snippets in Web search results where $\quad Q$ is issued as a query.

For example, when detecting hypernyms, the prefix is "<s> *are*" and the suffix is "*such as* <s>". If <s> is "whales", "*such as* whales" and "whales *are*" are issued to a Web search engine. We may find phrases "**mammals** *such as* whales" and "whales *are* **mammals**" in the search results. In this case, the term "**mammals**" satisfies all the conditions of being a member in the T.

If we only use one lexico-syntactic pattern, the process needs an advanced technique to extract appropriate related terms. For example, let us issue a query to detect hypernyms as follows.

$$\mathbf{Patterns} \quad := \quad \texttt{<t>} \parallel \textit{such as} \parallel \texttt{<s>}$$

Here, the definition of T will become

$$\begin{aligned} T = \{\texttt{<t>} \mid & \quad \mathbf{Patterns} \in Parts(\quad Q^{Pre}) \\ & \wedge \quad FirstNounPhrases\ (\texttt{<t>})\} \end{aligned}$$

where $FirstNounPhrases\ (\texttt{<t>})$ is a natural language processing function that finds a noun phrase in the text string, <t>, and POS tagging is necessary in the function. $FirstNounPhrases\ (\texttt{<t>})$ is just an example of an advanced technique, but a certain kind of processing should be prepared to extract accurate matches from the text.

Apart from this, as several methods [22]–[27] of calculating the relatedness between two terms have been proposed, we can use them to examine whether an extracted term is appropriate for a term related to the given term. As many of them used Web hit counts to calculate relatedness, it was necessary to access the Internet several times for each term found.

The proposed method, on the other hand, does not need any special processing, and it only accesses the Internet to obtain Web search results; consequently, it is both lightweight and fast. Let us explain our method in a little more detail.

B. Case of detection of coordinate terms

We proposed a method of detecting the coordinate terms of a given term in our past work [1]. This was an example of methods of high-speed detection of related terms using bi-directional lexico-syntactic patterns.

The approach to the detection of coordinate terms is based on the idea that the conjunction "*or*" connects two coordinate terms. If <s> and <t> are coordinate terms, both phrases should exist as

$$\begin{aligned} \mathbf{Patterns}^{Pre} \quad &:= \quad \texttt{<s>} \parallel \textit{or} \parallel \texttt{<t>} \\ \mathbf{Patterns}^{Post} \quad &:= \quad \texttt{<t>} \parallel \textit{or} \parallel \texttt{<s>} \end{aligned}$$

We try to find such phrases using Web search results to collect candidates for the coordinate terms of <s> and to confirm if a certain <t> is appropriate. The method consists of four steps.

Step 1 A query term is given.
Step 2 Two Web search queries are made.
Step 3 The Web search results (URLs, titles, and snippets) are obtained and analyzed.
Step 4 The candidates for coordinate terms are scored.

When a term is given, we make two Web queries according to the patterns. Where the given term is "Hillary Clinton", its queries are

- "Hillary Clinton *or*" and
- "*or* Hillary Clinton".

Each Web query is issued to a conventional Web search engine, and k search results are gathered for each. We usually use 100 as k. The queries are actually contained within double quotation marks for phrase searches that many conventional Web search engines support. That is, the double quotation marks are used to obtain text that contains whole phrases that are in the queries. Assume the titles and snippets in the search results contain the following sentences.

- "... that either Hillary Clinton or Barack Obama will emerge as the overwhelming favorite ...", and
- "... the voters lived in counties where Barack Obama or Hillary Clinton won by a landslide ...".

"Barack Obama" is one of the most appropriate coordinate terms for "Hillary Clinton". As both "Hillary Clinton *or* Barack Obama" and "Barack Obama *or* Hillary Clinton" are contained in the Web search results, "Barack Obama" is a member of T. The method regards "Barack Obama" as a coordinate term for "Hillary Clinton" in such cases.

As both of the bi-directional lexico-syntactic patterns consist of <s>, <t>, and "*or*", they are quite similar in this case, but we should regard them as completely different patterns. Using two such types of lexico-syntactic patterns, we can easily find appropriate cutting points for compound words.

That is, only "Barack Obama" is the text string that matches both the lexico-syntactic patterns in the two example sentences of "Hillary Clinton". There are actually many text strings that match either of the patterns. For example, in the first sentence,

- "Barack",
- "Barack Obama",
- "Barack Obama will",
- "Barack Obama will emerge", and
- "Barack Obama will emerge as"

match the $\mathbf{Patterns}^{Pre}$. In the second sentence,

- "Obama",

- “Barack Obama”,
- “where Barack Obama”,
- “counties where Barack Obama”, and
- “in counties where Barack Obama”

match the $\mathbf{Patterns}^{Post}$. All the wrong compound words only match either of the two patterns, and only the appropriate compound word “Barack Obama” matches both the patterns.

The method can find appropriate cutting points for compound words in this way even without morphological analysis. This is because it uses two different kinds of lexico-syntactic patterns.

Let us explain the method for English, where every word is separated by a space. However, Japanese and Chinese are languages with different styles in which words are not separated by a space. It is much more difficult to find appropriate cutting points for compound words in these languages than it is in English. Nevertheless, the proposed method can work quite well, even in such languages.

For example, as “*Ya*” in Japanese means “*or*” in English, bi-directional lexico-syntactic patterns as those in the following can be used to detect coordinate terms in Japanese.

$$\mathbf{Patterns}^{Pre} := \texttt{<s>} \parallel \underline{Ya} \parallel \texttt{<t>}$$
$$\mathbf{Patterns}^{Post} := \texttt{<t>} \parallel \underline{Ya} \parallel \texttt{<s>}$$

In the same way as in English, words that match both the patterns can be regarded as coordinate terms of a given term in Japanese.

C. Work flow in method

We explained the detection of coordinate terms as one specific case of the proposed method. To apply the method to many kinds of relationships between terms, the queries to a Web search engine should be changed so that they can be applied to many kinds of relationships between terms. However, it is occasionally difficult to find two different kinds of lexico-syntactic patterns that exist between <s> and <t>. The formula in Section III-A only assumes cases in which Web queries automatically determine lexico-syntactic patterns. Therefore, we extend the method where bi-directional lexico-syntactic patterns and Web queries can, to some degree, be independently determined.

The practical flow for the extended method is as follows.

1) <s> is given.
2) $\mathbf{Patterns}^{Pre}$ and $\mathbf{Patterns}^{Post}$ are prepared.
3) Web query Q^{Pre} that is for $\mathbf{Patterns}^{Pre}$ and Web query Q^{Post} that is for $\mathbf{Patterns}^{Post}$ are prepared.
4) Titles and snippets in the top k search results for Q^{Pre} are obtained from a Web search engine. $Parts(\ Q^{Pre})$ denotes a multiset of any phrase in the titles and snippets. $Parts(\ Q^{Post})$ for Q^{Post} are obtained in the same manner.
5) $Count^{Pre}(\mathbf{t}_i)$ denotes the cardinality in $Parts(\ Q^{Pre})$, which is the number of times a certain text string $\mathbf{t}_i$ appears in $Parts(\ Q^{Pre})$.
6) $Count^{Post}(\mathbf{t}_i)$ denotes the cardinality in $Parts(\ Q^{Post})$, which is the number of times a certain text string $\mathbf{t}_i$ appears in $Parts(\ Q^{Post})$.
7) A score for candidate text string $\mathbf{t}_i$ is calculated based on $Count^{Pre}(\mathbf{t}_i)$ and $Count^{Post}(\mathbf{t}_i)$. This is denoted by $Scor\ (\mathbf{t}_i)$.
8) Stop words are removed and if $Scor\ (\mathbf{t}_i)$ is greater than threshold θ, $\mathbf{t}_i$ is regarded as a term related to <s>.

First, term <s> is given. Bi-directional lexico-syntactic patterns $\mathbf{Patterns}^{Pre}$ and $\mathbf{Patterns}^{Post}$ are determined according to the kinds of related terms that are required. Although they are prepared taking <s> into consideration in many cases, it is not necessary for <s> to be contained in the patterns. The number of $\mathbf{Patterns}^{Pre}$ or $\mathbf{Patterns}^{Post}$ can be more than one.

Next, Web queries Q^{Pre} and Q^{Post} are made either automatically or manually. Both the Web queries should contain <s> even if <s> is not contained in the patterns.

The Web queries are issued to a Web search engine. Usually, a conventional Web search engine can be used, such as Google Web search[9], Yahoo! Web search[10], and Live Search[11]. In specific cases, other kinds of Web search engines are more suitable. For example, we created a system that indicated the change in coordinate term relationships over time [28]. A news archive search was used at that time.

In our implementation discussed in this paper, we used the Yahoo! search Web service API [12] as a Web search engine.

Because the proposed approach only uses titles and snippets in search results, the Internet is only accessed a few times. This means the results can be returned very quickly. The top 100 search results are usually sufficient to obtain five to ten related terms, and the processing time for this is about 3 to 5 sec.

The text obtained is cleaned, i.e., unnecessary marks such as quotation marks are removed and all letters are changed into lower case. Then a multiset of any phrase in the text is obtained. Actually, as text strings either appearing **after** the prefix or **before** the suffix can be regarded as sufficiently related terms, it is sufficient just to capture them. As we demonstrated in the example on “Barack Obama”, several different-length phrases match a pattern. We treat each phrase length individually. The minimum unit is a term in space in separated languages such as English, and a character is the minimum unit in Japanese and Chinese. In English, it is sufficient to extract a maximum of 10 terms. In Japanese, it is sufficient to extract a maximum of 15 characters.

[9]http://www.google.com/
[10]http://search.yahoo.com/
[11]http://www.live.com/
[12]http://developer.yahoo.co.jp/search/

Web search results occasionally contain the same sentences several times. In such cases, the same phrase is found in these sentences. If we count the number of appearances of a phrase that matches the patterns, these cases are individually counted. Consequently, it would be better if we counted the number of variations in sentences in which the phrase matched the patterns. They are counted for both $\mathbf{Patterns}^{Pre}$ and $\mathbf{Patterns}^{Post}$.

Finally, the found terms are scored. The main point with the proposed method is that a term that matches *both* patterns $\mathbf{Patterns}^{Pre}$ and $\mathbf{Patterns}^{Post}$ is an appropriate term related to a given term. This means that both $Count^{Pre}(\mathbf{t}_i)$ and $Count^{Post}(\mathbf{t}_i)$ should be more than zero. The following formula for scoring works well in many cases.

$$Scor\ (\mathbf{t}_i) \quad := \quad \sqrt{Count^{Pre}(\mathbf{t}_i) \cdot Count^{Post}(\mathbf{t}_i)}$$

This is the geometrical average of $Count^{Pre}(\mathbf{t}_i)$ and $Count^{Pre}(\mathbf{t}_i)$. If we use zero as the threshold, θ, terms that appear once or more in both the patterns are regarded as appropriately related terms. Finally, stop words and some meaningless terms are removed.

In this approach, we need to adjust four items according to what kinds of related words are required. These are $\mathbf{Patterns}^{Pre}$, $\mathbf{Patterns}^{Post}$, Q^{Pre}, and Q^{Post}. It is possible to quickly find various kinds of related terms when appropriate settings are found.

D. Examples of bi-directional lexico-syntactic patterns and Web queries

One of the settings that rapidly detect hypernyms is

$$\begin{aligned}
\mathbf{Patterns}^{Pre} &:= (\textit{typical} \mid \textit{famous}) \parallel \texttt{<t>} \\
\mathbf{Patterns}^{Post} &:= \texttt{<t>} \parallel \textit{such as} \parallel \texttt{<s>} \\
Q^{Pre} &:= \text{“}\texttt{<s>}\text{”} \wedge \\
&\quad (\text{“}\textit{typical}\text{”} \vee \text{“}\textit{famous}\text{”}) \\
Q^{Post} &:= \text{“}\textit{such as}\ \texttt{<s>}\text{”}
\end{aligned}$$

where some parts of the lexico-syntactic patterns are represented as regular expressions.

Although Q^{Post} is almost the same as $\mathbf{Patterns}^{Post}$, Q^{Pre} is related to but different from $\mathbf{Patterns}^{Pre}$. $\mathbf{Patterns}^{Pre}$ materially consists of multiple patterns. Thus, there can be more than one lexico-syntactic pattern. Moreover, $\mathbf{Patterns}^{Pre}$ does not contain <s>. Thus, patterns do not need to contain <s>.

Some example results for this detection of hypernyms are listed in Table I. The queries are “Sagrada Familia”, “Ronaldo”, “paella”, and “sangria”.

100 search results for each Web query were used. The geometrical average was used to calculate the scores for the found terms. The results indicate quite good precision.

Here, we compare the performances of the proposed method and the previous approach [2]. The proposed method supports richer expressions for the lexico-syntactic patterns than the previous one. Namely, the bi-directional lexico-syntactic patterns in the proposed method do not need to contain <s>, although the ones in the previous one should contain <s>.

TABLE I.
EXAMPLE RESULTS FOR DETECTION OF HYPERNYMS USING SETTING SUPPORTED BY PROPOSED METHOD.

Sagrada Familia	
Found Term	Score
places	2.4
works	2.2
sights	2.0
landmarks	2.0
architecture	2.0
sites	1.4
barcelona	1.4

Ronaldo	
Found Term	Score
player	11.6
football	3.0
soccer players	2.4
bootballers	1.0
star	1.0
soccer stars	1.0

paella	
Found Term	Score
dishes	7.9
recipes	3.5
Spanish dishes	3.2
dish	3.0
Spanish food	2.4
rice dishes	2.0
food	1.4

sangria	
Found Term	Score
drink	2.4
wine	2.2
summer	1.7
spanish wines	1.0

TABLE II.
EXAMPLE RESULTS FOR DETECTION OF HYPERNYMS USING SETTING SUPPORTED BY BOTH PROPOSED METHOD AND PREVIOUS METHODS.

Sagrada Familia	
Found Term	Score
N/A	-

Ronaldo	
Found Term	Score
legends	1.4

paella	
Found Term	Score
specialties	1.4

sangria	
Found Term	Score
wine	1.4

For example, the $\mathbf{Patterns}^{Pre}$ in the above example to detect hypernyms is not supported in the previous approach. The following setting can be used in both the proposed method and the previous methods.

$$\begin{aligned}
\mathbf{Patterns}^{Pre} &:= \texttt{<s>} \parallel \textit{are} \parallel \texttt{<t>} \\
\mathbf{Patterns}^{Post} &:= \texttt{<t>} \parallel \textit{such as} \parallel \texttt{<s>} \\
Q^{Pre} &:= \text{“}\texttt{<s>}\ \text{are”} \\
Q^{Post} &:= \text{“}\textit{such as}\ \texttt{<s>}\text{”}
\end{aligned}$$

The hypernyms can be detected by using this setting. The results for the same queries, i.e., “Sagrada Familia”, “Ronaldo”, “paella”, and “sangria”, are listed in Table II. No terms can be detected for “Sagrada Familia”. Only one term can be found for each of the other queries. The lexico-syntactic patterns in the setting are well-known and have been well-used to detect hypernyms [4]. They, however, seem to be too restricted for extracting hypernyms from listed text resources. Finally, comparing the results in Table I and II, we can see that the proposed method outperformed the previous one.

E. Evaluation of method

Here, we discuss the speed, the number of returned related terms, and the precision of the proposed approach.

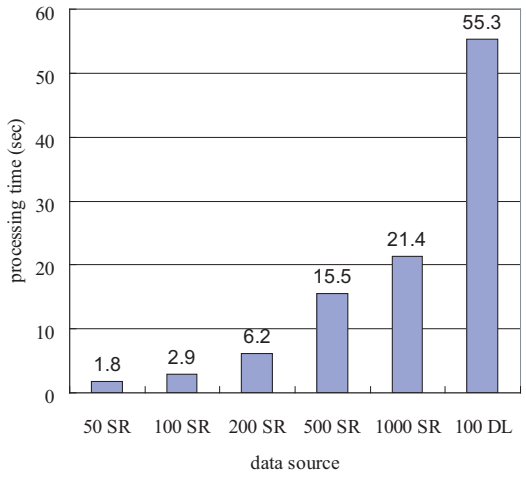

Figure 1. Average processing time.

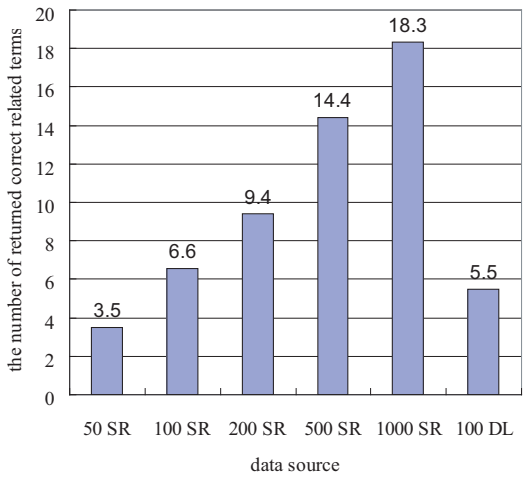

Figure 2. Average number of correct related returned terms.

Four different kinds of detection of related terms were prepared. They returned four kinds of related terms.

1) Hyponyms.
2) Coordinate terms.
3) Names of famous areas for a given products.
4) Names of World-heritages sites in a given area.

The geometrical average was used to calculate the scores of related terms, and the threshold, θ, was set to zero. We used five different k, i.e., 50, 100, 200, 500, and 1000. Additionally, we attempted a case where 100 actual Web pages were downloaded instead of using the titles and snippets in the search results. Three queries were issued in each of the five cases, and some related terms were returned for each of the three queries. We obtained the average processing time, the average number of correct related returned terms, and the average precision. All the experiments were conducted on a laptop PC (Intel(R) Pentium(R) M processor 2.26 GHz, 2 GB RAM, and Windows XP SP2) with an optical Internet connection.

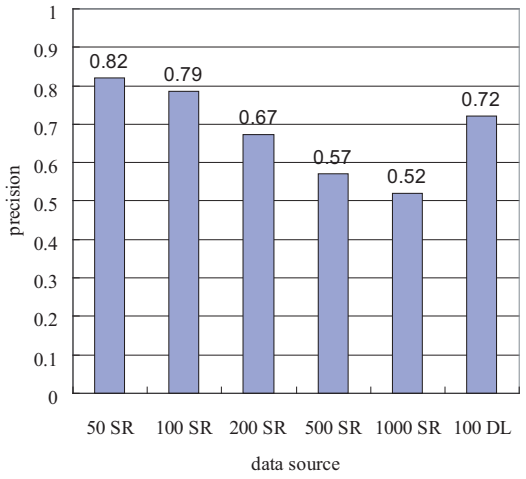

Figure 3. Average precision.

Figures 1, 2, and 3 are bar charts of the results. The labels from **50 SR** to **1000 SR** represent the five cases using 50 up to 1000 search results. The label **100 DL** indicates the case where 100 Web pages were downloaded.

From Figure 1, we can see that the less data the method used, the faster the result was returned. Even though the **100 DL** case downloaded Web pages in multithreads, it was much slower than **1000 SR**. Figure 2 shows the average number of correct related terms that were returned in each of the six cases, and Figure 3 shows their average precision. In the **100 SR** case, the precision remained around 80%, and the average number of related terms returned was more than six. Its average processing time was less than 3 sec. **100 SR** could be used in many services and applications to obtain information from the Web on demand.

F. Summary of method

This section describes the high-speed method used to detect many kinds of related terms using bi-directional lexico-syntactic patterns. To apply the method to a certain kind of relationship, we need to determine two different kinds of appropriate lexico-syntactic patterns. The first is $\mathbf{Patterns}^{Pre}$, starting with a prefix that occurs just **before** the target terms appear and the second is $\mathbf{Patterns}^{Post}$, ending with a suffix that occurs just **after** the target terms appear. Web queries Q^{Pre} and Q^{Post} can be automatically created, but it is also possible to determine these individually.

There are several other minor settings in the method. The first is for the number of Web search results obtained for each Web query. When 100 search results are obtained, several related terms can be found and the speed at which results are returned is sufficiently fast, as we stated in Section III-E. Another is for calculating the scores of candidate terms. The geometrical average is usually used as the score, and the threshold, θ, is zero.

IV. Automatic Discovery of Lexico-Syntactic Patterns

The proposed method works well if it is possible to find useful bi-directional lexico-syntactic patterns for detecting related terms in a certain relationship. However, it is sometimes difficult to find appropriate patterns. Here, we propose a method of automatically discovering useful lexico-syntactic patterns.

A user enters a pair of <s> and <t>. The method then discovers $\mathbf{Patterns}^{Pre}$, $\mathbf{Patterns}^{Post}$, Q^{Pre}, and Q^{Post}. For example, when we enter the term pair "Margaret Thatcher" and "the Iron Lady", the lexico-syntactic patterns and Web queries to find the person's nickname are automatically found.

A. Lexico-syntactic patterns and Web queries

Web queries with the method can be independently made from lexico-syntactic patterns, even though they are closely related. However, in automatically finding the settings, we uniquely determine the Web queries when lexico-syntactic patterns are determined. That is, $\mathbf{Patterns}^{Pre}$ determines Q^{Pre}, and $\mathbf{Patterns}^{Post}$ determines Q^{Post} in the same way as was described in Section III-A.

Let us assume lexico-syntactic patterns can be found as follows when the term pair "Sagrada Familia" and "Antoni Gaudi" is entered.

$$\mathbf{Patterns}^{Pre} := \texttt{<s>}\ \textit{designed by} \parallel \texttt{<t>}$$
$$\mathbf{Patterns}^{Post} := \texttt{<t>} \parallel \textit{was}$$

Then, Q^{Pre} and Q^{Post} are automatically determined as

$$Q^{Pre} := \text{"}\texttt{<s>}\ \textit{designed by}\text{"}$$
$$Q^{Post} := \text{"}\texttt{<s>}\text{"} \wedge \text{"}\textit{was}\text{"}.$$

The $\mathbf{Patterns}^{Pre}$ contains <s>. In this case, the Web query for the pattern is the pattern itself contained within double quotation marks.

However, the $\mathbf{Patterns}^{Post}$ does not contain <s>. In this case, the Web query is an *AND* query of <s> and the pattern itself, where both are enclosed within double quotation marks.

This makes the settings easier to find to automatically determine Web queries from lexico-syntactic patterns.

B. Candidates for lexico-syntactic patterns

The purpose is to find useful bi-directional lexico-syntactic patterns when a pair of <s> and <t> is entered.

First, 1000 search results are gathered where the Web query is <s> $\wedge$ <t>. The candidates for prefixes and suffixes are extracted from them.

For example, as useful text strings for the prefix should frequently appear *just before* <t>, text strings in such places should be extracted from the search results. The numbers of appearances of text strings are counted. For English, it is sufficient to extract a maximum of six terms.

TABLE III.
Preliminary candidates for suffixes for "Apple"-"Steve Jobs"

Candidates for suffixes	Count	Preliminary Score	
is	94	67	$(94-27)$
is the	27	21	$(27-6)$
is the CEO	6	0	$(6-6)$
is the CEO of	6	1	$(6-5)$
is the CEO of <s>	5	50	$((5-0)*10)$

For Japanese, the unit is not a term but a character, and to extract a maximum of 15 characters is sufficient to find useful patterns. After the text strings are extracted, some of them are removed such as those containing commas, stop words, or text strings that contain <s> at the midpoint.

Next, the extracted text strings are scored according to the number of times they appear.

For example, when the given <s>-<t> pair is "Apple"-"Steve Jobs", some of the text strings are extracted as suffixes, which appear *just after* "Steve Jobs", as listed in Table III.

The count in the table is the number of appearances, and the preliminary score is calculated as

$$Pr\ \ Scor\ \ (t_i) := Count(t_i) - \max_{t \in S_i}(Count(t))$$
$$S_i := \{s | s \in S_{all} \wedge s \text{ contains } t_i\}$$

where t_i is an extracted text string, $Pr\ \ Scor\ \ (t_i)$ is the preliminary score for t_i, S_{all} is a set of all the extracted text strings, and $Count(t)$ is the number of times the extracted text string t appears. The formula means that the score for t_i is calculated by subtracting the maximum number of appearances of t that contains t_i from the number of appearances of t_i.

In the above case, the preliminary score for "*is*" becomes 67 because "*is*" appears 94 times and "*is the*", which contains "*is*", appears 27 times. The preliminary score for "*is the*" becomes 21 and is calculated by subtracting 6, which is the number of times "*is the CEO* <s>" appears, from 27, which is the number of times "*is the*" appears. The preliminary score for "*is the CEO of* <s>" is 5 because text strings that contain <s> at the midpoint are removed; consequently, there are no longer text strings that contain "*is the CEO of* <s>".

Moreover, if an extracted text string contains <s>, its preliminary score is multiplied by 10 because usually patterns that contain <s> yield better results for the proposed method. Finally, the preliminary score for "*is the CEO of* <s>" becomes 50. These scored text strings are candidates for suffixes.

Candidates for prefixes are also obtained in the same manner. They should appear frequently *just after* <t>.

The leftmost column in Table IV lists some extracted candidates for bi-directional lexico-syntactic patterns whose preliminary scores are in the top ten when "Apple"-"Steve Jobs" is entered.

TABLE IV.
CANDIDATES FOR BI-DIRECTIONAL LEXICO-SYNTACTIC PATTERNS FOR "APPLE"-"STEVE JOBS"

Candidates for prefixes	$PreSc\ re$	<t> count	Hit counts
<s> CEO	830	60	4,390,000
CEO	120	52	82,400,000
<s> chief executive	90	43	678,000
<s> and	70	0	-
of	61	1	1,080,000,000
<s> founder	50	23	208,000
<s> inc chief executive	40	18	84,400
<s> chief	40	7	1,360,000
<s> thrive without	40	15	303
<s> or	30	0	-

Candidates for suffixes	$PreSc\ re$	<t> count	Hit counts
and <s>	180	0	-
is	67	0	-
announces <s>	60	9	132,000
and	58	0	-
is the CEO of <s>	50	33	5,620
and other <s>	40	8	908,000
has	40	1	612,000,000
of <s>	40	0	-
portrait made from <s>	30	27	100
is <s>	30	0	-

C. Examination of candidates

The candidates somehow represent the relationships between given <s> and <t>. However, this does not mean that they can work as useful lexico-syntactic patterns in the proposed method. Therefore, we examined candidates that had high preliminary scores.

There are several features that useful patterns should have. The most important is that <t> can be found many times when the pattern is used. A pattern is usually more useful if it can find <t> more frequently than the others. However, if a pattern detects <t> too many times, then it is not useful because it is too specific and cannot work except for the given <s>. Another important feature is that a certain number of Web search results can be obtained when the pattern is used as a Web query. The hit count should be more than 1000 because it is difficult to find many related terms if not enough text is gathered.

Table IV lists several candidates for patterns when "Apple"-"Steve Jobs" is entered. The <t> count is the number of <t>s that are found when the candidate is used as a pattern in the method. 100 search results are used for this. The table also lists the hit count for Web searches when the candidate is issued as a query.

For example, the suffix "*is the CEO of* <s>" can detect <t> 33 times and the hit count is also sufficiently large. Such a pattern is useful. For the prefix, "<s> *CEO*" is obviously a useful pattern.

The "*portrait made from* <s>" for $\mathbf{Patterns}^{Pre}$ or "<s> *thrive without*" for $\mathbf{Patterns}^{Post}$ can find <t> many times, but as there are not enough hit counts, they are not regarded as useful patterns.

Where "PC"-"Panasonic" is entered, the method finds a candidate for a suffix as

$$\text{suffix} \quad := \quad \text{toughbook}$$

There were 7,840,000 hits for the pattern, which is quite good. However, the pattern found "Panasonic" 89 times when <s> was "PC", which is too high. The pattern is too specific for "Panasonic", and cannot work for any other term.

In actual use, we chose a pattern that could find <t> the most number of times, but less than 70, and also one that had more than 1,000 hits.

D. Results of automatic discovery of lexico-syntactic patterns

We will now present several examples where lexico-syntactic patterns are automatically discovered. We will also present related terms that are detected using these.

The lexico-syntactic patterns found from "Apple"-"Steve Jobs" are

$$\mathbf{Patterns}^{Pre} \quad := \quad \texttt{<s>} \parallel \textit{CEO} \parallel \texttt{<t>}$$
$$\mathbf{Patterns}^{Post} \quad := \quad \texttt{<t>} \parallel \textit{is the CEO of} \parallel \texttt{<s>}$$

Some results using the patterns are listed in Table V. Almost all of the found terms are correct except for Terry Semel in Yahoo! because he has resigned from being the CEO.

The patterns found for "Ganymede"-"Jupiter" are as follows.

$$\mathbf{Patterns}^{Pre} \quad := \quad \textit{satellite of} \parallel \texttt{<t>}$$
$$\mathbf{Patterns}^{Post} \quad := \quad \texttt{<t>} \parallel \textit{and its}$$

This is used to obtain the planet's name from its satellite. Some results using the patterns are listed in Table VI. Each top scored term is a correct answer. Even though there are several mistakes, the score for the correct answers is much greater than for the incorrect terms.

TABLE V.
DETECTION OF RELATED TERMS BASED ON RELATIONSHIPS SUCH AS "APPLE"-"STEVE JOBS".

FedEX	
Found term	Score
Fred Smith	7.9
Smith	5.0

Microsoft	
Found term	Score
Steve Ballmer	29.4
Ballmer	13.4
Steve	13.2

Nissan	
Found term	Score
Carlos Ghosn	12.4
Ghosn	8.9
Carlos	6.8

Yahoo	
Found term	Score
Jerry Yang	13.0
Terry	6.0
Yang	5.5
Terry Semel	5.5
Semel	2.8

TABLE VI.
DETECTION OF RELATED TERMS BASED ON RELATIONSHIPS SUCH AS "GANYMEDE"-"JUPITER".

Phobos	
Found term	Score
Mars	44.4
Jupiter	3.7
the planet	3.0
the planet Mars	1.4

Moon	
Found term	Score
Earth	13.8
the Earth	8.5
the planet	4.7
Jupiter	4.2
Saturn	2.4
planet	1.4

Titan	
Found term	Score
Saturn	38.1
the planet	5.3

Europa	
Found term	Score
Jupiter	22.4
Europe	3.5

The patterns found for "France"-"Paris" are as follows.

$$\mathbf{Patterns}^{Pre} := \textit{hotel in} \parallel \texttt{<t>}$$
$$\mathbf{Patterns}^{Post} := \texttt{<t>} \parallel \textit{is the capital of} \texttt{<s>}$$

This setting obtains the name of the capital when the country's name is entered. Some example results using the patterns are listed in Table VII. All the results have a correct answer in the results. Only the result for "Australia" does not have a correct answer at the top. The other top results are correct and the scores are greater than those for the other incorrect terms. We did not think of manually using *hotel in* as a prefix, but it actually worked very well.

Thus, the proposed method of discovering useful lexico-syntactic patterns can work for many kinds of relationships.

V. CONCLUSIONS

We proposed a method of detecting ontological knowledge from the Web that could obtain terms related to a given term. Knowledge was extracted from snippets and titles in Web search results, and at that time two different kinds of lexico-syntactic patterns were used. Because bi-directional lexico-syntactic patterns allow the method to find appropriate cutting points for extracted words, it can return results very quickly and its precision is excellent.

We also developed a method of automatically discovering bi-directional lexico-syntactic patterns as it is sometimes difficult to find appropriate patterns to detect related terms in a certain relationship. When a pair of terms is given, the method finds patterns that detect related terms in the relationship between the given terms.

TABLE VII.
DETECTION OF RELATED TERMS BASED ON RELATIONSHIPS SUCH AS "FRANCE"-"PARIS".

Spain	
Found term	Score
Madrid	21.0
Barcelona	3.3

Canada	
Found term	Score
Ottawa	12.5
Toronto	4.0

Japan	
Found term	Score
Tokyo	21.9
Kyoto	2.0
Osaka	1.7

Australia	
Found term	Score
Sydney	15.0
Canberra	8.8
Melbourne	3.2
Perth	2.0
Brisbane	1.0
Darwin	1.0

North Korea	
Found term	Score
Pyongyang	14.4
Seoul	2.0

South Korea	
Found term	Score
South	31.2
a city	1.4

ACKNOWLEDGMENT

This work was supported in part by the following projects and institutions: Grants-in-Aid for Scientific Research (Nos. 18049041 and 21700105) from MEXT of Japan, a Kyoto University Global COE Program entitled "Informatics Education and Research for Knowledge-Circulating Society," and the National Institute of Information and Communications Technology, Japan.

REFERENCES

[1] H. Ohshima, S. Oyama, and K. Tanaka, "Searching coordinate terms with their context from the Web," in *Proc. of the 7th International Conference on Web Information Systems Engineering (WISE 2006)*, October 2006, pp. 40–47.

[2] H. Ohshima and K. Tanaka, "Real time extraction of related terms by bi-directional lexico-syntactic patterns from the Web," in *Proc. of the 3rd International Conference on Ubiquitous Information Management and Communication (ICUIMC2009)*, January 2009, pp. 441–449.

[3] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller, "Introduction to WordNet: An on-line lexical database," *International Journal of Lexicography 3(4)*, pp. 235–312, 1990.

[4] M. A. Hearst, "Automatic acquisition of hyponyms from large text corpora," in *Proc. of the 14th International Conference on Computational Linguistics (COLING 1992)*, August 1992, pp. 539–545.

[5] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates, "Web-scale information extraction in knowitall: (preliminary results)," in *Proc. of the the 13th international conference on World Wide Web (WWW 2004)*, May 2004, pp. 100–110.

[6] Z. Ghahramani and K. Heller, "Bayesian sets," in *Proc. of the 19th Annual Conference on Neural Information Processing Systems (NIPS 2005)*, 2005.

[7] D. Lin, "Automatic retrieval and clustering of similar words," in *Proc. of the 36th annual meeting on Association for Computational Linguistics*, 1998, pp. 768–774.

[8] K. Shinzato and K. Torisawa, "A simple www-based method for semantic word class acquisition," in *Proc. of the Recent Advances in Natural Language Processing (RANLP 2005)*, 2005, pp. 493–500.
[9] K. Shinzato and K. Torisawa, "Acquiring hyponymy relations from Web documents," in *Proc. of Human Language Technology Conference/North American chapter of the Association for Computational Linguistics annual meeting (HLT-NAACL 2004)*, 2004, pp. 73–80.
[10] M. Yamaguchi, H. Ohshima, S. Oyama, and K. Tanaka, "Unsupervised discovery of coordinate terms for multiple aspects from search engine query logs," in *Proc. of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence (WI 2008)*, December 2008 (to appear).
[11] M. Komachi and H. Suzuki, "Minimally supervised learning of semantic knowledge from query logs," in *Proc. of the 3rd International Joint Conference on Natural Language Processing (IJCNLP 2008)*, January 2008, pp. 358–365.
[12] M. Paşca, "Weakly-supervised discovery of named entities using web search queries," in *Proc. of the 16th ACM Conference on Information and Knowledge Management (CIKM 2007)*, November 2007, pp. 683–690.
[13] M. Paşca and B. V. Durme, "What you seek is what you get: Extraction of class attributes from query logs," in *Proc. of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007)*, February 2007, pp. 2832–2837.
[14] S. Sekine and H. Suzuki, "Acquiring ontological knowledge from query logs," in *Proc. of the 16th International Conference on World Wide Web (WWW 2007)*, May 2007, pp. 1223–1224.
[15] M. Sanderson and B. Croft, "Deriving concept hierarchies from text," in *Proc. of the 22nd ACM SIGIR Conference (SIGIR '99)*, 1999, pp. 206–213.
[16] E. Glover, D. M. Pennock, S. Lawrence, and R. Krovetz, "Inferring hierarchical descriptions," in *Proc. of the 11th International Conference on Information and Knowledge Management (CIKM 2002)*, 2002, pp. 507–514.
[17] S. Oyama and K. Tanaka, "Query modification by discovering topic from Web page structures," in *Proc. of the 6th Asia Pacific Web Conference (APWEB 2004)*, 2004, pp. 553–564.
[18] K. Nakayama, T. Hara, and S. Nishio, "A thesaurus construction method from large scaleweb dictionaries," in *Proc. of the 21st International Conference on Advanced Networking and Applications (AINA 2007)*, May 2007, pp. 932–939.
[19] K. Nakayama, T. Hara, and S. Nishio, "Wikipedia mining for an association web thesaurus construction," in *Proc. of the 8th International Conference on Web Information Systems Engineering (WISE 2007)*, December 2007, pp. 322–334.
[20] T. Hokama and H. Kitagawa, "Extracting mnemonic names of people from the Web," in *Proc. of the 9th International Conference on Asian Digital Libraries (ICADL 2006)*, November 2006, pp. 121–130.
[21] K. W. Church and P. Hanks, "Word association norms, mutual information, and lexicography," in *Proc. of the 27th Annual Meeting of the Association for Computational Linguistics*, 1998, pp. 76–83.
[22] P. D. Turney, "Mining the Web for synonyms: PMI-IR versus LSA on TOEFL," in *Proc. of the 12th European Conference on Machine Lerning (ECML 2001)*, September 2001, pp. 491–502.
[23] M. Baroni and S. Bisi, "Using cooccurrence statistics and the Web to discover synonyms in a technical language," in *Proc. of the 4th International Conference on Language Resources and Evaluation (LREC 2004)*, May 2004, pp. 1725–1728.
[24] R. L. Cilibras and P. M. Vitanyi, "The Google similarity distance," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 3, pp. 370–383, March 2007.
[25] P. D. Turney, "Measuring semantic similarity by latent relational analysis," in *Proc. of the 19th International Joint Conference on Artificial Intelligence (IJCAI 2005)*, July 2005, pp. 1136–1141.
[26] P. D. Turney and M. L. Littman, "Corpus-based learning of analogies and semantic relations," *Machine Learning*, vol. 60, pp. 251–278, August 2005.
[27] D. Bollegala, Y. Matsuo, and M. Ishizuka, "WWW sits the SAT-Measuring relational similarity on the Web," in *Proc. of the 18th European Conference on Artificial Intelligence (ECAI 2008)*, July 2008, pp. 333–337.
[28] H. Ohshima, A. Jatowt, S. Oyama, and K. Tanaka, "Visualizing changes in coordinate terms over time: An example of mining repositories of temporal data through their search interfaces," in *Proc. of the International Workshop on Information-explosion and Next Generation Search (INGS 2008)*, April 2008.

Hiroaki Ohshima has been an assistant professor of the Graduate School of Informatics, Kyoto University since 2006. He received the BS and MS degrees in Engineering from Kobe University, in 2000 and 2003, respectively. In 2006, he received the PhD degree in Informatics from Kyoto University. His research interests include Web search and mining, information retrieval, and database systems. He is a member of the ACM, the Information Processing Society of Japan (IPSJ), the Institute of Electronics, Information and Communication Engineers (IEICE), and the Database Society of Japan (DBSJ).

Katsumi Tanaka received the BS, MS and PhD degrees in Information Science from Kyoto University, in 1974, 1976 and 1981, respectively. In 1986, he joined the Department of Instrumentation Engineering, faculty of Engineering at Kobe University, as an associate professor. In 1994, he became a full professor at the Department of Computer and Systems Engineering Department, Faculty of Engineering, Kobe University. Since 2001, he has been a professor of the Graduate School of Informatics, Kyoto University. His research interests include database theory and systems, Web search and mining, and multimedia content retrieval. Dr. Tanaka is a member of the ACM, IEEE, the Database Society of Japan (DBSJ) and the Information Processing Society of Japan (IPSJ). He is currently a vice president of DBSJ and the fellow of IPSJ.