Journal of Information Processing Vol.20 No.1 216–227 (Jan. 2012)

[DOI: 10.2197/ipsjjip.vol.20.216]

Regular Paper

TSUBAKI: An Open Search Engine Infrastructure for Developing Information Access Methodology

Keiji Shinzato[1,†1,a)] Tomohide Shibata[1] Daisuke Kawahara[1] Sadao Kurohashi[1]

Received: April 11, 2011, Accepted: September 12, 2011

Abstract: Due to the explosive growth in the amount of information in the last decade, it is getting extremely harder to obtain necessary information by conventional information access methods. Hence, creation of drastically new technology is needed. For developing such new technology, search engine infrastructures are required. Although the existing search engine APIs can be regarded as such infrastructures, these APIs have several restrictions such as a limit on the number of API calls. To help the development of new technology, we are running an open search engine infrastructure, TSUBAKI, on a high-performance computing environment. In this paper, we describe TSUBAKI infrastructure.

Keywords: search engine infrastructure, web-based data sharing, natural language processing, natural language search, info-plosion

1. Introduction

There is a huge amount of information being published on the World Wide Web (WWW) today, as signified by the popular term, information explosion, and this information spans a multitude of forms including news articles, encyclopedia articles, know-how documentation, and word-of-mouth information from individuals. The keyword and page-rank-based type of search currently available is inadequate to use all of this information on the WWW effectively, so it is becoming increasingly important to find ways to express information needs more effectively, to organize and summarize the information on the WWW. In particular, the following techniques are potentially helpful.

- Performing searches based on user requirements expressed in natural-language sentences.
- Automatically changing search-result rankings to match the user's ideas or plans.
- Organizing concepts related to a given topic and providing an overall understanding of the topic.
- Examining the distribution of opinions on a given topic and classifying them in terms such as minority or majority.
- Detecting the reliability or inconsistencies within the results of a search.

In order to implement these techniques, a search engine infrastructure is needed. Currently there are several APIs available which provide access to results from commercial search engines, but there are problems with using these as a base for research and development.

(1) Use is limited, as in the number of API calls or number of documents in the search result.
(2) The indexes are updated frequently, so search results cannot be reproduced.
(3) The metrics used for ranking results are not made public.

Thus, we have built and are operating the TSUBAKI[*1] open search-engine infrastructure to resolve these problems. TSUBAKI is a search engine oriented to research use. It covers approximately one hundred million Japanese Web pages and provides users with transparent, reproducible search results. An API is also provided, with no limitations such as accesses per day or number of search results[*2].

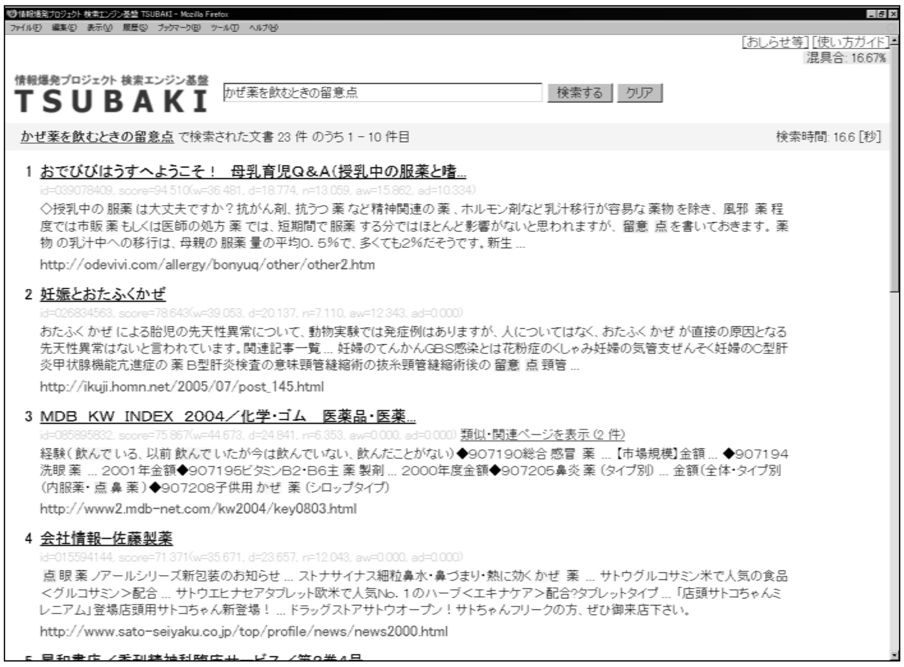
TSUBAKI also has the following features:

- Manages large-scale Web pages using Web standard format.
- Provides flexible and accurate searches through use of synonyms and dependency relations.

Web standard format is a common XML format designed for sharing the results of Web page analysis. Research has been making more progress into ways for extracting knowledge from Web pages and other tasks related to material on the WWW, but when actually doing research using Web pages, there are many troublesome processing steps that must be faced before getting to the research. Specifically, the large set of pages must be crawled, the Japanese pages must be extracted from the crawl result, and sentences must be extracted from these pages. Taking just sentence extraction as an example, periods can be used to extract sentences from a corpus such as a newspaper article, but the division into sentences is often unclear on Web pages, where HTML tags are used and unconventional usage such as smiley faces (e.g.: (^-^)) and "(笑)" to express feelings are common. For these reasons, sentence segmentation is a somewhat messy process, but it is fundamental, and its performance can have a great effect on later language analysis or application performance. Thus, it is very important to gather a large number of Web pages to use as

1 Graduate School of Informatics, Kyoto University, Kyoto 606–8501, Japan
†1 Presently with Rakuten Institute of Technology
a) shinzato@i.kyoto-u.ac.jp

*1 http://tsubaki.ixnlp.nii.ac.jp/
*2 TSUBAKI may not support unlimited query submitting when the number of users is increased. At that time, the submitted query will be pushed into a query queue, and be processed later.


情報爆発プロジェクト 検索エンジン基盤 TSUBAKI - Mozilla Firefox
ファイル(F) 編集(E) 表示(V) 履歴(S) ブックマーク(B) ツール(T) ヘルプ(H)
[おしらせ等] [使い方ガイド]
混具合: 16.67%
情報爆発プロジェクト 検索エンジン基盤
TSUBAKI
かぜ薬を飲むときの留意点
検索する
クリア

かぜ薬を飲むときの留意点 で検索された文書 23 件 のうち 1 – 10 件目　　検索時間: 16.6 [秒]

1 おでびびはうすへようこそ！　母乳育児Q&A(授乳中の服薬と嗜...
id=039078409, score=94.510(w=36.481, d=18.774, n=13.059, aw=15.862, ad=10.334)
◇授乳中の 服薬 は大丈夫ですか？抗がん剤、抗うつ 薬 など精神関連の 薬 、ホルモン剤など乳汁移行が容易な 薬物 を除き、風邪 薬 程度では市販 薬 もしくは医師の処方 薬 では、短期間で 服薬 する分ではほとんど影響がないと思われますが、留意 点 を書いておきます。薬物 の乳汁中への移行は、母親の 服薬 量の平均0．5%で、多くても2%だそうです。新生 ...
http://odevivi.com/allergy/bonyuq/other/other2.htm

2 妊娠とおたふくかぜ
id=026834563, score=78.643(w=39.053, d=20.137, n=7.110, aw=12.343, ad=0.000)
おたふく かぜ による胎児の先天性異常について、動物実験では発症例はありますが、人についてはなく、おたふく かぜ が直接の原因となる先天性異常はないと言われています。関連記事一覧 ... 妊婦のてんかんGBS感染とは花粉症のくしゃみ妊婦の気管支ぜんそく妊婦のC型肝炎甲状腺機能亢進症の 薬 B型肝炎検査の意味頸管縫縮術の抜糸頸管縫縮術後の 留意 点 頸管 ...
http://ikuji.homn.net/2005/07/post_145.html

3 MDB KW INDEX 2004／化学・ゴム　医薬品・医薬...
id=085895832, score=75.867(w=44.673, d=24.841, n=6.353, aw=0.000, ad=0.000) 類似・関連ページを表示 (2 件)
経験（飲んで いる、以前 飲んで いたが今は飲んでいない、飲んだことがない)◆907190総合 感冒 薬 ... 【市場規模】金額 ... ◆907194洗眼 薬 ... 2001年金額◆907195ビタミンB2・B6主 薬 製剤 ... 2000年度金額◆907205鼻炎 薬 (タイプ別) ... 金額(全体・タイプ別(内服薬・ 点 鼻 薬）◆907208子供用 かぜ 薬 (シロップタイプ)
http://www2.mdb-net.com/kw2004/key0803.html

4 会社情報–佐藤製薬
id=015594144, score=71.371(w=35.671, d=23.657, n=12.043, aw=0.000, ad=0.000)
点 眼 薬 ノアールシリーズ新包装のお知らせ ... ストナサイナス細粒鼻水・鼻づまり・熱に効く かぜ 薬 ... サトウグルコサミン米で人気の食品<グルコサミン>配合 ... サトウエヒナセアタブレット欧米で人気No. 1のハーブ<エキナケア>配合?タブレットタイプ ... 「店頭サトコちゃんミレニアム」登場店頭用サトコちゃん新登場！ ... ドラッグストアサトウオープン！サトちゃんフリークの方、ぜひ御来店下さい。
http://www.sato-seiyaku.co.jp/top/profile/news/news2000.html

Fig. 1 Example of the search result (query: かぜ薬を飲むときの留意点 (Points to note when taking cold medicine)).

a standard, to perform the preprocessing required for research as described above, and to share this collection in helping to improve the usability of Web pages as a language resource. With this in mind, TSUBAKI provides a search space covering 100 million Japanese Web pages, and publishes the data with appropriate preprocessing, such as sentence extraction, already completed. This data is managed in the Web standard format.

Another feature of TSUBAKI is its use of synonymous expressions and dependency relations to provide flexible and accurate searches. In addition to words, synonymous expressions and dependency relations are also registered in the index, which helps to absorb expression mismatches and achieves searches that emphasize the dependency relations between words. **Figure 1** shows the results of searching for "かぜ薬を飲むときの留意点" (Points to note when taking cold medicine-also includes mixed use of kanji and hiragana) using TSUBAKI. The search result includes pages with terms like "風邪薬" (kanji for hiragana word in the search string) and "服薬" (a synonym for "薬を飲む", or taking medicine).

The remainder of the paper is organized as follows. We first describe the Web standard format, and then explain natural-language analysis and indexing components of the TSUBAKI search-engine infrastructure in Sections 3 and 4, respectively. In Section 5, we discuss the types of search conditions and constraints that can be specified with TSUBAKI, and the methods used for document scoring and generating snippets. We also described the computing environment and the search flow used by TSUBAKI. Finally, we describe how to utilize TSUBAKI APIs in Section 6.

2. Web Standard Format

Web standard format is an XML format designed for sharing the results of Web page analysis. In this section, we give an overview of the Web standard format and discuss conversion of Web pages to this format as well as handling of large-scale Web standard format data sets.

2.1 Data Format

A Web standard format data (hereafter standard format data) is constructed from a Web page, and includes information about the Web page that is frequently used for research, such as the page title and URL, link information, the Japanese text, and results of analyzing the text. This information can be accessed easily using conventional XML-document search modules.

Thus, all available information of a Web page is centralized by a standard format data. This allows TSUBAKI users to obtain information which they want without accessing databases corresponding to each information.

Table 1 gives a definition of the standard format tag set, and **Fig. 2** gives an example of standard format data.

2.2 Conversion to Web Standard Format

Conversion of a Web page to standard format data involves the following four steps.

(1) Japanese page detection
(2) Meta and link information extraction
(3) Sentence extraction
(4) Linguistic analysis of the text

Step (2) consists of simply extracting information from `<META>` and `<A>` tags, so we will discuss only steps (1) and (3) here. Step (4) will be discussed in Section 3.

2.2.1 Japanese Page Detection

We determine if the page is Japanese text by using a method similar to that of Kawahara et al. [1], based on the percentage of Japanese particles in the page. Specifically, if the character encoding specified in the `<META>` tag is one of euc-jp, x-euc-jp, iso-2022-jp, shift jis, windows-932, x-sjis, or shiftjp, and the rate of occurrence of the Japanese particles "が", "を", "に", "は", "の",

Table 1 Tags defined in the standard format.

Tag name	Description	Child elements	Attributes
`<StandardFormat>`	Standard format root tag.	`<Header>`, `<Text>`	**Url:** URL of the source Web page. **OriginalEncoding:** Encoding of the source Web page. **Time:** Data creation timestamp (YYYY-MM-DD hh:mm:ss format).
`<Header>`	Tag containing header metadata.	`<Title>`, `<InLinks>`, `<OutLinks>`, `<Keywords>`, `<Description>`	None.
`<Title>`	Title of source page.	`<RawString>`, `<Annotation>`	**Offset:** Offset from beginning of file. **Length:** Title length (bytes). **Id:** Sentence ID.
`<InLinks>`	A collection of links to the source page.	`<InLink>`	None.
`<InLink>`	A single link to the source page.	`<RawString>`, `<Annotation>`, `<DocIDs>`	None.
`<OutLinks>`	A collection of links out of the source page.	`<OutLink>`	None.
`<OutLink>`	A single link from the source page.	`<RawString>`, `<Annotation>`, `<DocIDs>`	None.
`<DocIDs>`	A collection of document IDs.	`<DocID>`	None.
`<DocID>`	A single document ID.	A nine-digit numerical document ID.	None.
`<Keywords>`	The list of keywords given in the <meta name="Keywords" content="〜〜〜"> tag of the source page.	`<RawString>`, `<Annotation>`	**Offset:** Offset from the beginning of the file. **Length:** Keywords length (bytes).
`<Description>`	Description given in the <meta name= "Description" content="〜〜〜"> tag of the source page.	`<RawString>`, `<Annotation>`	**Offset:** Offset from the beginning of the file. **Length:** Description length (bytes).
`<Text>`	Tag containing text of the source page.	`<S>`	None.
`<S>`	Text of the source page.	`<RawString>`, `<Annotation>`	**Offset:** Offset from the beginning of the file. **Length:** Source length (bytes). **Id:** Sentence ID.
`<RawString>`	Tag containing the text of the document as a single sentence.	Text of the document extracted as a single sentence.	None.
`<Annotation>`	Tag containing the results of analyzing a single sentence.	Results of analysis using various tools.	**Scheme:** Name of tool used for analysis (e.g. Juman, Knp, SynGraph, etc.).

and "で" ("ga", "wo", "ni", "ha", "no", and "de") is more than 0.5%, the page is determined to be Japanese text. If the character encoding is not specified in the `<META>` tag, a guess of the encoding is done using the encoding() function in the `Encode::guess` module included in Perl 5.8.

2.2.2 Sentence Extraction

The text in the page is divided into paragraphs based on clues such as the location of HTML block tags and consecutive end-of-line markers. Then paragraphs are divided into sentences using punctuation characters ("。", "?", "!", "♪", "…", "・・・", etc.), face marks ((^-^)) and characters used to express emotion ("(笑)" (smile), "(汗)" (sweat/stress), etc.). When doing so, if the character or notation is enclosed in parenthesis, then it is not considered to be a sentence delimiter.

Within Web pages there are sections where multiple anchor texts follow one-another consecutively (**Fig. 3**). When sentences are extracted from this sort of page, all of the anchor text is extracted as one sentence in a continuous string. This is not always appropriate, so when there are consecutive anchor texts, we first determine whether joining them is appropriate and then join them only if appropriate. Whether anchor text should be joined is calculated using document frequency for compound nouns extracted from 100 million Web pages. Specifically, expressions with document frequency of ten or more are considered suitable to be joined. Using this method, sentences can be extracted appropriately even from Web pages like that shown in Fig. 3.

Some sentences are over-splitted by this partitioning process, so the following rules are used to join sentence segments back together. Here, S_i is used to indicate a segment being joined, and S_{i+1} is the segment immediately following S_i.

Rule1: If an odd number of quotes (") are included in S_i and S_{i+1}, then they are joined.

Rule2: If S_i begins with text indicating an itemized form, such as "A.", S_i and S_{i+1} are joined.

Rule3: If S_{i+1} begins with characters such as "と", "っ" or "です", and S_i ends with a closing parenthesis, "!" or "?", they are joined.

Finally, processing to form itemized lists is done. The itemized list is expressed in text form as shown in **Fig. 4**. For this type of page, each element of the list is initially extracted as an individual sentence.

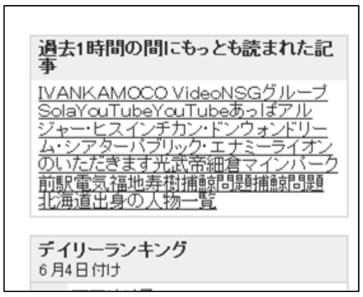
Sentence 1: The following:
Sentence 2: 1. Kiyomizuji Temple
Sentence 3: 2. Kinkakuji Temple
Sentence 4: 3. Ryoanji Temple
Sentence 5: are my favorite tourist sites.

Itemized lists in text form are identified using certain rules, to prevent extracting list elements as individual sentences, and the sentences within the identified ranges are joined. Specifically, if a sentence ends in a period or comma, and is followed by sentences starting with characters that are often used for list items (e.g.: "A." or "1."), these sentences (1 to 4 above) are joined, together with the sentence immediately following the list (Sentence 5 above). In this way, the following sentence is extracted from the page in the figure.

Sentence 1: The following: 1. Kiyomizuji Temple 2. Kinkakuji Temple 3. Ryoanji Temple are my favorite tourist sites.

```
<?xml version="1.0" encoding="UTF-8"?>
<StandardFormat Url="http://www.kantei.go.jp/jp/koizumiprofile/1_sinnen.html" OriginalEncoding=
"Shift_JIS" Time="2006-08-14 19:48:51">
<Header>
  <Title Offset="21" Length="39" Id="0">
    <RawString>小泉総理プロフィール・信念</RawString>
  </Title>
  <OutLinks>
    <OutLink>
      <RawString>トップ</RawString>
      <DocIDs><DocID Url="www.kantei.go.jp/index.html">060936437</DocID></DocIDs>
    </OutLink>
  </OutLinks>
  <InLinks>
    <InLink>
      <RawString>小泉総理の信念</RawString>
      <DocIDs><DocID Url="http://mocuromi365.yh.land.to/">067985366</DocID></DocIDs>
    </InLink>
  </InLinks>
</Header>
<Text>
<S Id="1" Length="70" Offset="525">
  <RawString>小泉総理の好きな格言のひとつに「無信不立 (信無くば立たず)」があります。</RawString>
  <Annotation Scheme="SynGraph">
    <![CDATA[* 1D <文頭><サ変><人名><助詞><連体修飾><体言><係:ノ格><区切:0-4>
小泉 こいずみ 小泉 名詞 6 人名 5 * 0 * 0 NIL <文頭><漢字><かな漢字><名詞相当語><自立><タグ単位始><文節始><固有キー>
!! 0 1D <見出し:小泉>
  ! 0 <SYNID:小泉/こいずみ><スコア:1>
 ...
ます ます ます 接尾辞 14 動詞性接尾辞 7 動詞性接尾辞ます型 31 基本形 2 NIL <表現文末><かな漢字><ひらがな><活用語><付
属><非独立無意味接尾辞>
。 。 。 特殊 1 句点 1 * 0 * 0 NIL <文末><英記号><記号><付属>
!! 11 -1D <見出し:あります>
  ! 11 <SYNID:有る/ある><スコア:1>
  ! 11 <SYNID:s10957:有る/ある><スコア:0.99>
EOS]]></Annotation>
</S>
<S Id="2" Length="160" Offset="595">
  <RawString>論語の下篇「顔淵」の言葉で，弟子の子貢（しこう）が政治について尋ねたところ，孔子は「食料を
十分にし軍備を十分にして，人民には信頼を持たせることだ」と答えました。</RawString>
  <Annotation Scheme="SynGraph">
    <![CDATA[* 1D <文頭><助詞><連体修飾><体言><係:ノ格><区切:0-4>
論 ろん 論 名詞 6 普通名詞 1 * 0 * 0 "漢字読み:音 代表表記:論" <漢字読み:音><代表表記:論><文頭><漢字><かな漢字><名詞
相当語><自立><タグ単位始><文節始>
!! 0 1D <見出し:論>
  ! 0 <SYNID:論/ろん><スコア:1>
 ...
ました ました ます 接尾辞 14 動詞性接尾辞 7 動詞性接尾辞ます型 31 タ形 5 NIL <表現文末><かな漢字><ひらがな><活用語>
<付属><非独立無意味接尾辞>
。 。 。 特殊 1 句点 1 * 0 * 0 NIL <文末><英記号><記号><付属>
!! 21 -1D <見出し:答えました>
  ! 21 <SYNID:答える/こたえる><スコア:1>
  ! 21 <SYNID:s3095:応じる/おうじる><スコア:0.99>
EOS]]></Annotation>
</S>
...
</Text>
</StandardFormat>
```

Fig. 2 Example of a standard-format data.



過去1時間の間にもっとも読まれた記事

IVANKAMOCO VideoNSGグループSolaYouTubeYouTubeあっぱアルジャー・ヒスインチカン・ドンウォンドリーム・シアターパブリック・エナミーライオンのいただきます光武帝細倉マインパーク前駅電気福地寿樹捕鯨問題捕鯨問題北海道出身の人物一覧

デイリーランキング
6月4日付け

Fig. 3 Example of a section where multiple anchor texts follow one-another consecutively.

```
The following:

1. Kiyomizuji Temple

2. Kinkakuji Temple

3. Ryoanji Temple

are my favorite tourist sites.
```

Fig. 4 Example of an itemized list in a Web page.

Table 2 Precision and recall of the sentence splitter.

Precision (%)	Recall (%)
98.5 (197/200)	93.4 (197/211)

2.3 Converting a Large Set of Web Pages to Web Standard Format

Approximately 100 million Japanese pages were converted to Web standard format. These pages were taken from pages crawled by the Knowledge Clustered Group at the National Institute of Information and Communications Technology (NiCT) at 2010. Then, the Japanese pages were selected as described in Section 2.2. Each page was given a nine-digit numeric document ID, and both the original and the standard format page were given this number.

The computing environment used to perform the conversion included 162 machines with quad Intel Xeon 3.0 GHz CPUs and 4 GB of memory, operating in parallel using GXP2 [2] to perform the conversion. As a result, conversion of the 100 million Japanese pages to Web standard format took approximately four weeks to complete. These 100 million pages contained roughly 6 billion sentences, and the linguistic analysis described in Section 3, including morphological analysis, syntactic analysis and synonymous-expression alignment, was carried out on these sentences. The data size of the Web pages themselves was 1.4 TB, while the standard format data was 12 TB. Both are sizes after compression with gzip.

This Web standard format data is available from TSUBAKI through the APIs provided.

2.4 Evaluation of the Sentence Splitter

We randomly picked up 200 pairs of adjacent sentences from the standard format data collection, and then checked them to find whether a boundary between sentences in each pair was correct or not. We also checked whether each sentence in the 200 pairs included a missing boundary, a position that should be a sentence boundary and that is not recognized as a sentence boundary by the sentence splitter.

As a result, we found three over-splitted boundaries, and 11 missing boundaries from three sentences. This means that 211 sentence boundaries are included in the picked up sentences, and the sentence splitter correctly detects 197 sentence boundaries. **Table 2** shows the precision and recall of the sentence splitter. We can see that both the precision and recall can be achieved high performance.

Examples of sentences which have an over-splitted boundary are shown in **Fig. 5**. The symbol "！" is a cause of over-splitting because this symbol is used as a clue for sentence splitting. To avoid this type of problem, we need a list which consists of expressions that include a symbol used as a sentence boundary. The other over-splitting problem is derived from a <BR> tag. The current sentence splitter basically regards a <BR> tag as a sentence boundary. To solve this type of problem, the splitter needs to leverage context information such as sentences appearing on the both sides of a <BR> tag.

On the other hand, examples of sentences that contain missing boundaries are shown in **Fig. 6**. We need a list of expressions that

Pair A:
Sentence 1: お客様が入力する個人情報を含むお問い合わせデータは、Ｙａｈｏｏ！
Sentence 2: ＪＡＰＡＮを経由して当該物件の情報提供元に転送されます。

Pair B:
Sentence 3: 涼しいどころではなく、汗だくになりながら
Sentence 4: 二人の子供＋保護者で、８種目に参加しました

Fig. 5 Examples of over-splitted boundaries.

Sentence 5: 【送料無料】お客様の声から誕生した２０色羽根布団８点セットのカバー |【送料無料】２０色…
Sentence 6: トテーイ革命 | 佐久間Ｄ | 谷桃子 | 中田敦彦 | ＨＩＴＳ ＬＩＭＩＴＥＤ | 谷桃子ＳＰ付コンプリートセット価格：９，４５０円（税込、送料別）|【値段交渉お気軽にどうぞ】|【宅配便送料無料】|［新品］| 新品！

Fig. 6 Examples of missing boundaries. The symbol "|" is a position of a missing boundary.

can be regarded as sentence boundaries such as【送料無料】to divide a sentence like Sentence 5. Although Sentence 6 is not an anchor text, we can basically take the same approach with anchor text processing described in Section 2.2.2 if we can recognize the sentence as a series of (compound) nouns.

3. Natural Language Analysis

In this section, we discus the natural language analysis applied to sentences extracted in Section 2.2 and to search queries. First, morphological and syntactic analysis is applied to the sentences and search queries. The Japanese morphological analyzer JUMAN [3] and the Japanese syntactic parser KNP [4] were used. Through the JUMAN analysis, words that can be written in various ways, such as "こども", "子ども", and "子供" (various writings of kodomo, or child) are given a representative form ID. Within the index data, the representative form rather than the form that it appears in the document is used. This helps to reduce search leakage due to spelling variations.

After morphological and syntactic analysis, links are made among synonymous words and phrases. This is called synonymous expression alignment below. In this process, synonym relationships extracted automatically from ordinary dictionaries and from the Web corpus are used [5]. Handling the relationship between two synonymous expressions as a simple, two-item relationship results in combinatorial explosion, so we have used the SynGraph data structure proposed by Shibata et al. [5]. The SynGraph data structure expresses these relationships effectively by assigning expressions that have a synonym relationship with the same ID (hereafter, the synonymous expression ID). For example, a single synonymous expression ID would be assigned to all of the expressions "子供", "児童", and "稚児" (roughly: child, juvenile, and infant). This process prevents enumerating each of the relationship combinations.

Figure 7 shows the results of analyzing the expression "かぜ薬を飲むときの留意点" (Points to note when taking cold medicine). The words in the white boxes are the original text, while the words in the grey boxes are synonymous expression IDs assigned through the synonymous expression alignment process (e.g., the <服用> synonymous expression ID has been assigned to the "薬を飲む" phrase). The arrows represent dependency relations between phrases. The number appearing below

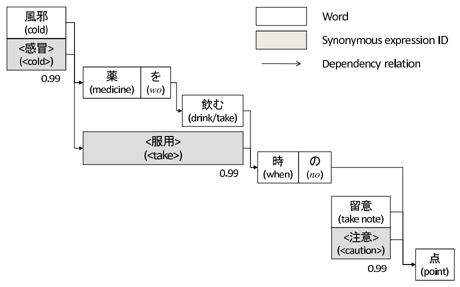


Fig. 7 Results of analyzing the text "かぜ薬を飲むときの留意点" (Points to note when taking cold medicine).

each synonymous expression ID is a score for the amount of semantic difference from the original expression, and is used for the frequency of appearance of the synonymous expression ID when creating the index data.

4. Indexing

TSUBAKI index data was created for the 100 million Japanese Web pages discussed in Section 2.3, using the results of linguistic analysis embedded in the Web standard format data. The index data is structured an inverted index with terms for representative forms, synonymous expression IDs, and dependency relations. When storing index data, dependency relations are handled as two-item relationships between the content words, without considering case markers. For example, for the phrase "かぜ薬を飲むときの留意点" in Fig. 7, the following expressions from the analysis results are registered in the index.

Word, Synonymous expression ID: 風邪, 薬, を, 飲む, 時, の, 留意, 点, <感冒>, <服用>, <注意>

Dependency relations: 風邪→薬, 薬→飲む, 飲む→時, 時→点, 留意→点

Note particularly that due to the representative form assigned by JUMAN, " かぜ " is extracted as "風邪" (hiragana and kanji rendering of "cold"). Further, the synonymous expression ID, <服用>, is also assigned to words like "服用" and "服薬" (in addition to "薬を飲む", or "take medicine"), so by registering <服用> in the index data, pages related to "薬を飲む" but can be found by using the other expression such as 服薬 and 服用.

Table 3 shows the statistics of indexed documents. Words, synonymous expressions and dependency relations are registered in index data with # of docs, Doc. ID, Freq. and positions. The # of docs indicates the number of documents including an indexed item. This number is used for loading information of documents that include the item from the index data. The Doc ID is the ID of an indexed document. The Freq. is the frequency of words, synonymous expression IDs and dependency relations in a document. The frequencies registered for words and dependency relations are actual frequencies of appearance in the documents. However, for the synonymous expression IDs, a score indicating the amount of difference from the original expression is used for the appearance frequency. For example, the synonymous expression ID, <感冒> for "風邪" is handled as appearing 0.99 times.

In order to support searches that take positions of words into account, such as phrase searches and proximity searches, the position of the word (in number of words from the beginning of the page), in addition to the frequency, is stored in the index data.

Table 3 Statistics of dataset. Document length is represented by the number of words in a web page.

Statistics	Value
# of words	81,634,647,061
# of sentences	8,764,858,879
# of documents	100,149,341
Average document length	815.12

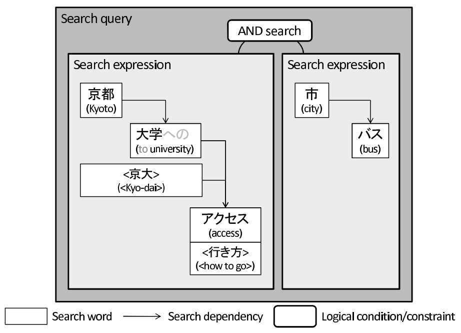


Fig. 8 Structure of the search query, "京都大学へのアクセス 市バス" ("Kyoto University Access City bus") (<xxx> indicates a synonymous expression ID).

TSUBAKI partitions this index data and scans it in parallel in order to increase search speed. More specifically, the index data for 100 million pages is partitioned in units of million-pages, and these are searched in parallel using 100 CPU cores.

5. Search

TSUBAKI supports queries expressed with keywords as well as natural-language sentence queries. We discuss search in this section, but before doing so we will define some terms. **Figure 8** shows a search query using AND, and the two phrases "京都大学へのアクセス" (Access to Kyoto University) and "市バス" (City Bus). In this paper, we call each of the white-space-separated strings search expressions ("京都大学へのアクセス" and "市バス" in the example). Search expressions are subjected to linguistic analysis as described in Section 3, and words, synonymous expression IDs and dependency relations are extracted from the results. Hereafter we will refer to these words and synonymous expression IDs as search words, and to these dependency relations as search dependency relations. In general, functional words are not used as search words, but in certain conditions they may be (e.g., a search word within a phrase constraint could be a functional word).

Search can be divided broadly into document gathering and document scoring. Basically, document gathering is done based on the search words, and document scoring uses both search words as well as search dependency relations.

In Section 5.1, we discuss the types of search conditions and constraints that can be specified with TSUBAKI, and in Section 5.2, we discuss the method used for document scoring. In Section 5.3 we discuss the method used for generating snippets, and evaluate the search method in Section 5.4. Finally, in Section 5.5, we describe the computing environment and the search flow used by TSUBAKI.

5.1 Search Conditions and Constraints

With TSUBAKI, constraints can also be established for each

Table 4 Search constraints supported by TSUBAKI.

Constraint name	Tag name	Description	Example
AND constraint	˜AND	Result must include all words in the search expression.	京都大学˜AND
OR constraint	˜OR	Result must contain one or more of the words in search expression.	京都大学˜OR
Phrase constraint	No tag. Enclosed in quotes (“ ”).	Same as the phrase search used by current search engines.	“京都大学”
Dependency relation constraint	˜FD	Result must contain all dependency relations in the search expression.	京都大学˜FD
Proximity constraint (words)	˜*N*W	Result must contain the words in the search expression in the order given and within a window of *N* words.	京都大学˜20W
Order-independent proximity constraint (words)	˜*N*w	Result must contain the words in the search expression within a window of *N* words (regardless of order).	京都大学˜20w

search expression, in addition to logical conditions between search expressions. Here, *constraints* refers to conditions other than logical conditions, such as whether to include the dependency relations with the search expression, or the size of the window within which the words in the search expression must appear. Constraints are expressed by appending a tilde (˜) and the constraint expression to the end of the search expression. **Table 4** shows the constraints that can be specified.

Combinations of search expressions with constraints specified may also be used. For example, the search query: “市バス” 京都大学へのアクセス˜20W, will result in a search for pages including the phrase “市バス” (City bus) as well as the words “京都” (Kyoto), “大学” (University), and “アクセス” (Access), in order and within a 20 word window.

If no explicit search conditions or constraints are specified, the logical condition defaults to `AND`, and the constraint defaults to `˜100w`.

5.2 Document Scoring

After gathering documents based on the search conditions (and constraints), a score of conformance to the search query is computed for each document. The documents are sorted in order of decreasing conformance and finally presented to the user. Okapi BM25 [6] is used to compute relevance to the query for each document. The words in the query and the document are usually used with Okapi BM25 to compute the relevance, but here we have extended the computation to also use dependency relations in computing relevance. If $T_{q_{word}}$ is the set of search words in query q, and $T_{q_{dpnd}}$ is the set of search dependency relations, then the relevance of document d to search query q is computed according to the following equation.

$$R(q,d) = (1-\beta)\sum_{t\in T_{q_{word}}} BM_{25}(t,d) + \beta \sum_{t\in T_{q_{dpnd}}} BM_{25}(t,d)$$

Here, β is a parameter that is used to regulate the extent to which dependency relations are used in the scoring, and we use $\beta = 0.2$. This value was decided experimentally using a test set created at the NTCIR3, 4 workshops [7], [8].

$BM_{25}(t,d)$ is defined by the following equation.

$$BM_{25}(t,d) = w \times \frac{(k_1+1)F_{dt}}{K+F_{dt}} \times \frac{(k_3+1)F_{qt}}{k_3+F_{qt}}$$

$$w = \log\frac{N-n+0.5}{n+0.5}, K = k_1\left((1-b)+b\frac{l_d}{l_{ave}}\right)$$

```
<TOPIC>
<NUM> 0008 </NUM>
<TITLE CASE="b"> Salsa, learn, methods </TITLE>
<DESC> I want to find out about methods for
learning how to dance the salsa </DESC>
:
</TOPIC>
```

Fig. 9 Example of a search topic defined in the NTCIR.

Here, F_{dt} is the frequency with which t appears in document d, F_{qt} is the frequency that t appears in q, N is the number of documents being searched, n is the document frequency of q, l_d is the length of document d (words), and l_{ave} is the average document length. Finally, k_1, k_3, and b, are Okapi parameters, for which we use values $k_1 = 1$, $k_3 = 0$ and $b = 0.6$.

5.3 Method for Generating Snippets

A set of ranges of W [*3] or fewer words and containing the words and synonymous expression IDs in the search query Q are found, a snippet is generated using the smallest range W_{min}. If multiple W_{min} are found, the range closest to the ending of the document is used.

We refer to the sentences in the range W_{min} as S_W. If S_W consists of K or many words, S_W is regarded as a snippet. On the other hand, if S_W includes fewer words than K, S_W extends a word to the right and left directions up to K words. We use the value of $K = 100$.

5.4 Evaluation of Search Performance

We evaluated the search performance using the test collections built at NTCIR-3 and NTCIR-4. These share a target document set, which consists of 11,038,720 Japanese Web pages. For the evaluation, we used 127 informational topics defined in the test collections (47 topics from NTCIR-3 and 80 topics from NTCIR-4). The example of the topic definition is shown in **Fig. 9**. `<TITLE>` includes a few keywords and `<DESC>` includes a single sentence, and they are queries reflecting user's information need. Note that the single sentence can be regarded as a natural language query. We separately exploited elements of `<TITLE>` and `<DESC>` for assessing the effectiveness of synonyms and dependency relations when giving a few keywords or a sentence as a query.

[*3] This number is specified as a proximity constraint in a query.

Table 5 Effectiveness of synonyms and dependency relations. Syn. means synonym, Dep. means dependency relations, and √ means a feature is activated. By default, TSUBAKI retrieves documents using synonyms and dependency relations (i.e., Syn. = √, Dep. = √). The highest value in each evaluation measure is emphasized by boldface. This does not mean statistical significance.

Query type	Syn.	Dep.	MRR	Prec@10	DCG@10	DCG@100	R-prec	MAP
Keyword			0.319	0.198	2.368	6.068	0.141	0.085
	√		0.325	0.209	2.475	6.268	0.137	0.085
		√	**0.350**	**0.212**	**2.592**	6.626	**0.145**	**0.091**
	√	√	0.343	**0.212**	2.575	**6.635**	**0.145**	**0.091**
Sentence			0.293	0.150	1.937	3.404	0.091	0.057
	√		0.293	0.151	1.945	3.374	0.083	0.055
		√	**0.308**	0.161	2.087	3.790	**0.100**	**0.065**
	√	√	**0.308**	**0.164**	**2.088**	**3.844**	0.097	**0.065**

The relevance of each document with respect to a topic was judged as *highly relevant*, *relevant*, *partially relevant*, *irrelevant* or *unjudged*. We regarded the highly relevant, relevant and partially relevant documents as *correct* answers, and then assessed search performance according to the following measures.

MRR: Reciprocal of the rank at which the first correct answer was found in the top 10 documents. If the top 10 documents did not contain the correct answer, we used 0 as the reciprocal rank.

Prec@10: Rate of the correct answers in the top 10 documents.

DCG@*N*: Discounted cumulative gain (DCG) [9] is an evaluation measure that takes advantage of multi graded relevance. The DCG at a rank N is defined as:

$$DCG@N = rel_1 + \sum_{i=2}^{N} \frac{rel_i}{\log_2 i},$$

where rel_i is the graded relevance of a document at a rank i.

R-prec: Precision after R documents have been retrieved where R is the number of the correct answers for a topic.

MAP: Average precision which is the average of precisions after each correct document is retrieved in the top 1,000 documents.

Table 5 shows the evaluation results. The results showed that search performance was improved by synonyms and dependency relations.

5.5 Machine Environment

The TSUBAKI machine environment and search process are shown in **Fig. 10**. TSUBAKI conducts searches using four types of servers: a load-balance server, search master servers which perform search-query analysis, search-slave servers which perform document gathering and scoring, and snippet servers. Document gathering and scoring uses M machines with m parallelism and snippet generation uses P machines with p parallelism. We actually used $M = 50$, $m = 2$, $P = 20$, and $p = 2$. With this computing environment, generating 1,000 search results for a single query requires an average of about 10 seconds.

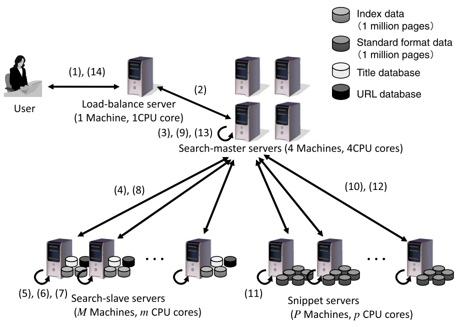
The role of each type of server is described below.

Load-balance server: Queries received from users are forwarded to the search-master server with the lowest load. This server is implemented using the Load Balancer associated with Apache 2.3.

Search-master servers: Analyze the search queries forwarded to them by the load-balance server, and forwards the results to the search-slave servers. They then gather the results returned by the search-slave servers and generate a search-result string as shown in Fig. 1. In doing so, they send requests to the snippet servers to get the required snippets.

Search-slave servers: Gather documents using the index data, based on the results of analyzing the search query, as forwarded to them by a search-master server. Then, using the scoring method discussed in Section 5.2, compute a relevance value for each document.

Snippet servers: Generate snippets using the linguistic analysis results in the standard-format data and the results of analyzing the search query. For increasing speed of the generation, the standard-format data are installed in the local storage of each snippet server.


Index data (1 million pages)
Standard format data (1 million pages)
Title database
URL database
User
(1), (14)
Load-balance server (1 Machine, 1CPU core)
(2)
(3), (9), (13)
Search-master servers (4 Machines, 4CPU cores)
(4), (8)
(10), (12)
(5), (6), (7)
Search-slave servers (M Machines, m CPU cores)
(11)
Snippet servers (P Machines, p CPU cores)

(1) Query Q is sent to the load-balance server.
(2) Query Q is sent to the master-search server with the lowest load.
(3) Query Q is analyzed linguistically and search conditions, C, search words, Q_{word} and search dependency relations, Q_{dpnd}, are extracted.
(4) C, Q_{word} and Q_{dpnd} are sent to the search-slave servers.
(5) Pages matching C, Q_{word} and Q_{dpnd} are gathered.
(6) The relevance of each document, D, to search query, Q, is computed.
(7) The title and URL for each page is retrieved from the database.
(8) The pages matching the search query, along with title, URL and relevancy are returned to the search-master server.
(9) Results from each search-slave server are merged and sorted according to relevance.
(10) The document numbers from the R highest-ranked documents and the search query are sent to the snippet servers.
(11) Snippets are generated by the snippet servers.
(12) Snippets are returned to the search-master server.
(13) A search result screen is generated (see Fig. 1).
(14) The search result is presented to the user.

Fig. 10 TSUBAKI computing environment and search process.

6. TSUBAKI API

TSUBAKI publishes an API conforming to REST principles [10] which allows searches to be conducted specifying the search conditions and constraints discussed in Section 5.1. The request URL for accessing this API is as follows:

- tsubaki.ixnlp.nii.ac.jp/api.cgi

Table 6 Request parameters available with the TSUBAKI API.

Parameter	Value	Description
query	string	A URL-encoded utf-8 string containing the search query. Required to obtain a search result.
start	integer	Rank of the first search result to be retrieved.
results	integer	Number of search results to retrieve. Default is 10.
logical_operator	AND/OR	Search logical operator. Default is AND.
only_hitcount	0/1	Set to 1 to retrieve only the hit count. Set to 0 to retrieve the actual results. Default is 0.
force_dpnd	0/1	Set to 1 for documents containing all dependencies in the query, and 0 for otherwise. Default value is 0.
snippets	0/1	Set to 1 if snippets are required, 0 otherwise. Default is 0.
near	integer	Perform a proximity search, with the condition that words in the query appear within n words of each other. The order of words in the query and as they appear is taken into consideration.
id	string	Document ID required to retrieve individual documents. Required when retrieving documents in original Web format or in standard format.
format	html/xml	Specifies whether to retrieve a document in original Web format or in standard format. Required when id is specified.

Search results can be obtained by sending an HTTP request appending the request parameters listed in **Table 6** to this URL. For example, to retrieve the top 20 results for the search expression “京都の観光名所” (Famous Kyoto site-seeing spots), send a request with the following URL.

- http://tsubaki.ixnlp.nii.ac.jp/api.cgi?query=%E4%BA%AC %E9%83%BD%E3%81%AE%E8%A6%B3%E5%85%89 %E5%90%8D%E6%89%80&start=1&results=20

The meaning of the query, start, and results parameters specified in this request URL are as follows:

query: URL encoded string containing keywords (“京都の観光名所” in this example).

start: The rank (offset) of the first result desired from among all search results.

results: The number of results desired, starting with start.

The results of the above query are shown in **Fig. 11**. The results are given with markup tags in XML format. The tag definitions are given in **Table 7**.

The latency of TSUBAKI API is shown in **Table 8**. We measured the latency according to the following settings:

(1) Get only hitcount
(2) Get top 100 search results without snippets
(3) Get top 1,000 search results without snippets
(4) Get top 100 search results with snippets
(5) Get top 1,000 search results with snippets

The values in Table 8 are the average values of 50 Japanese search queries. The example of the used queries is shown in **Table 9**. Note that the API can accept natural language queries such as “かぜ薬を飲むときの留意点” (Points to note when taking cold medicine) shown in Fig. 1.

We next discuss how to obtain the Japanese Web pages and the standard format data as discussed in Section 2. To retrieve this data, access the API specifying the format and id parameter in the request URL. For example, if the document ID is 012345678, the URL used to retrieve the Web page would be as follows:

- http://tsubaki.ixnlp.nii.ac.jp/api.cgi?format=html& id=012345678

Similarly, to obtain the standard format data, change the value of format to xml:

- http://tsubaki.ixnlp.nii.ac.jp/api.cgi?format=xml& id=012345678

Finally, we introduce the use case of the API. TSUBAKI is used as a search engine infrastructure on some online systems such as WISDOM[*4]. These online systems request the API for collecting documents related to a user query and standard format data corresponding to the collected documents. By collecting standard format data, these systems do not need to linguistically analyze documents in a search result. In other words, these

Table 7 Tags and attributes included in search results.

Field	Description
ResultSet	This tag encloses the search results, and it has the following attributes: time: Date and time when the search was conducted. query: Search expressions. totalResultsAvailable: Number of documents matching a search query. totalResultsReturned: Number of documents returned. firstResultPosition: The rank (offset) of the first result document returned, as specified in the search query. logicalOperator: The logical operator specified for the search. dpnd: Whether to consider dependency relations in the search. 0: Search without considering dependency relations. 1: Search with dependency relations. filterSimpages: Whether the similar-page filter is used or not. 0: Similar-page filter is not used. 1: Similar-page filter is used.
Result	This tag encloses the information regarding a single result document. It has the following attributes. ID: Document ID. Score: Document score. The document ID is necessary to retrieve the cached Web page or the standard format data for the page.
Title	Page title
Url	Page URL
Snippet	Sentences in the page that are relevant to the search query.
Cache	Information related to cached pages.
Url	URL of cached page.

Table 8 The latency of the TSUBAKI API.

Setting	Time [sec.]
Only hitcount	3.37
top 100 results (without snippets)	3.54
top 1,000 results (without snippets)	5.40
top 100 results	4.50
top 1,000 results	10.1

Table 9 Example of Japanese queries used for measurement of the API latency.

アガリクス (Agaricus), バイオエタノール (Bioethanol), ダイエット食品 (Diet food), 携帯電話電磁波 (Microwave of cell-phone), 原子力発電 (Nuclear), Ｗｉｎｄｏｗｓ Ｖｉｓｔａ (Windows Vista), ステロイド剤 (Steroid)

[*4] http://wisdom-nict.jp/

```
<ResultSet time="2009-02-08 23:22:39" query="京都の観光名所" totalResultsAvailable="7262"
totalResultsReturned ="20" firstResultPosition="1" logicalOperator="AND" forceDpnd="0" dpnd="1"
 anchor="1" filterSimpages="1" sort_by="score">
<Result Rank="1">
  <Title>Ｄｉｇｉｓｔｙｌｅ京都ＭＡＰ｜京都の観光名所、京都の名店をＧｏｏｇｌｅＭａｐで検索</Title>
  <Url>http://www.digistyle-kyoto.com/map/</Url>
  <Snippet>京都の観光・地域情報満載！京都ファンの為の情報サイト。Ｄｉｇｉｓｔｙｌｅ京都（デジスタイル京都）
の掲載店舗や観光名所をＧｏｏｇｌｅＭａｐで簡単検索。Ｄｉｇｉｓｔｙｌｅ京都ＭＡＰ｜京都の観光名所、京都の名
店をＧｏｏｇｌｅＭａｐで検索京都の観光・地域情報満載！京都ファンの為の... Ｄｉｇｉｓｔｙｌｅ京都（デジスタ
イル京都）の掲載店舗や観光名所をＧｏｏｇｌｅＭａｐで簡単検索。</Snippet>
  <Cache>
     <Url>http://tsubaki.ixnlp.nii.ac.jp/index.cgi?cache=006304506&KEYS=%E4%BA%AC%E9%83%BD%2F%E3%
81%8D%E3%82%87%E3%81%86%E3%81%A8%2Cs4283%3A%E8%A6%B3%E5%85%89%2F%E3%81%8B%E3%82%93%E3%81%93%E3%81
%86%3B%E8%A6%B3%E5%85%89%2F%E3%81%8B%E3%82%93%E3%81%93%E3%81%86%2Cs636%3A%E6%99%AF%E5%8B%9D%2F%E3
%81%91%E3%81%84%E3%81%97%E3%82%87%E3%81%86%3B%E5%90%8D%E6%89%80%2F%E3%82%81%E3%81%84%E3%81%97%E3%
82%87</Url>
  </Cache>
</Result>
...
<Result Rank="20">
  <Title>ＫＫＳブログ：修学旅行生向けの新観光名所？　京都で大学を見学してみる</Title>
  <Url>http://www.kknews.co.jp/wb/archives/2007/05/post_818.html</Url>
  <Snippet>全国自作視聴覚教材コンクール募集... トップページへ　｜絵本ギャラリー「モダニズムの絵本日常の中の
芸術」　?●【教育ニュース】修学旅行生向けの新観光名所？京都で大学を見学してみる　（２００７年０５月０８日）
大学コンソーシアム京都は、サイトで「京の「学び」道案内」というコンテンツを公開している。</Snippet>
  <Cache>
     <Url>http://tsubaki.ixnlp.nii.ac.jp/index.cgi?cache=005449025&KEYS=%E4%BA%AC%E9%83%BD%2F%E3%
81%8D%E3%82%87%E3%81%86%E3%81%A8%2Cs4283%3A%E8%A6%B3%E5%85%89%2F%E3%81%8B%E3%82%93%E3%81%93%E3%81
%86%3B%E8%A6%B3%E5%85%89%2F%E3%81%8B%E3%82%93%E3%81%93%E3%81%86%2Cs636%3A%E6%99%AF%E5%8B%9D%2F%E3
%81%91%E3%81%84%E3%81%97%E3%82%87%E3%81%86%3B%E5%90%8D%E6%89%80%2F%E3%82%81%E3%81%84%E3%81%97%E3%
82%87</Url>
  </Cache>
</Result>
</ResultSet>
```

Fig. 11 Example of the search result obtained using TSUBAKI API.

systems can more quickly exploit linguistically rich information such as dependency relations than they analyze documents themselves. This is a big advantage for online systems using the rich information.

We show examples of online systems that utilize TSUBAKI as a search engine infrastructure.

WISDOM [11]: A system that helps its users to judge the credibility of information on the Web by providing various information such as information sender and major and contradicting statements in documents related to a given query.

Statement Map [12]: A system that allows its users to easily grasp relations, such as synonymous and antonymous relations, between statements described in documents in a search result.

WebClustering [13], [14]: A system that organizes and summarizes Web information. This system creates clusters of documents including a user query, and then summarizes each cluster.

7. Related Work

Table 10 shows the results of comparing the TSUBAKI API with APIs from commercial search engines. APIs published by existing commercial search-engines have limits such as the number of queries allowed per day or the number of results that can be retrieved, which presents problems when attempting to use these systems as a base for system development or research. Furthermore, the index data for these commercial search engines is updated frequently, so it is difficult to obtain reproducible search results. This means that it is very difficult to compare results with leading research [15] using the number of search-engine hits returned, or the search results themselves. The search algorithms and ranking methods used by the commercial search engines are also not made public, so it is not possible to know from the user side, exactly what is being done when a search is conducted. As a result, it is very difficult to properly verify the effectiveness of techniques used by a search engine.

Table 10 Comparison of APIs from commercial search engines and TSUBAKI (as of Feb. 2010).

Feature	Google	Yahoo!	TSUBAKI
Number of API requests permitted per day	No limit	50,000	No limit
Number of URLs in a search result	1,000	1,000	No limit
Cached pages provided	Yes	Yes	Yes
Page analysis results provided	No	No	Yes
Index data is updated	Yes	Yes	No

As an open search engine infrastructure, our research can also be related to the various open source search engine projects, such as Indri [*5], Nutch [*6] and Rast [*7]. These projects focused on development of open-source modules, and they are not operating search engines using these modules. This is different from our research. The comparison between TSUBAKI and open source projects with respect to indexing and ranking measures are listed in **Table 11**. In addition to words, TSUBAKI registers synonyms and dependency relations with the index data. This is the feature

[*5] http://www.lemurproject.org/indri/
[*6] http://lucene.apache.org/nutch/docs/en/
[*7] http://projects.netlab.jp/rast/

Table 11 Comparison with indexing and ranking measure.

Search Engine	Indexing	Ranking Measure
TSUBAKI	word, synonym, dependency relation	OKAPI BM25
Indri	word*	TF·IDF, OKAPI BM25, Language Model based score, and others
Nutch	character bi-gram, word	TF·IDF
Rast	character bi-gram, word	TF·IDF

*) Indri can index annotations such as the output of a part-of-speech tagger if users prepare the annotation data.

of our research.

In terms of dealing with Web pages on a large scale, this research is related to the Web Laboratory project [16]. This project is jointly supported by Cornell University and the Internet Archive, and provides the Web pages stored in the Internet Archive together with Web page analysis tools for the purpose of supporting research related to the Internet. The Web Laboratory is currently in alpha release, but access to the Web pages and the set of tools is only available to persons related to Cornell University.

The Wikia project is also conducting related research, in that they are developing and operating a search engine provided with a transparent search algorithm. However, the core of the search algorithm has not yet been made public.

There are information retrieval systems which take advantage of natural language processing. Popescu et al. [17] developed a system which converts a given natural language query into an SQL sentence using synonyms and the parsing result of the query. Miyao et al. [18] proposed a system for retrieving biomedical correlations from MEDLINE. For retrieving biomedical correlations precisely, their system exploits predicate-argument structures and ontological databases. Powerset [*8] is a search engine based on natural language processing. With Powerset, in addition to keyword searches, natural language sentence and synonymous expression searches can be done. However, this search engine only applies to documents in the English Wikipedia site, and general Web pages cannot be searched.

8. Conclusion

In this paper we have discussed the open search-engine infrastructure called TSUBAKI. TSUBAKI allows anyone to freely obtain search results covering approximately 100 million Japanese Web pages through a public API. It features (1) Web pages stored and shared in Web standard format, and (2) flexible and accurate searches taking synonymous expressions and dependency relations into consideration. The documentation on TSUBAKI is available at http://tsubaki.ixnlp.nii.ac.jp/api.html.

The TSUBAKI API can be used in a variety of situations, and already it has been used in the following:

- To obtain large-scale corpus with syntactic analysis for knowledge acquisition.
- To compute the strength of co-occurrences of word pairs based on hit counts in order to obtain related terms.
- To obtain the pages to be clustered in a search-result clustering system.
- To obtain the pages to be analyzed for helping users to judge the credibility of information.

In the future, we plan to improve the system as a search engine infrastructure by consolidating the hardware environment and software to allow a larger number of users to use the system, to provide faster searches and to increase the robustness of the system.

*8 http://www.powerset.com/

Reference

[1] Kawahara, D. and Kurohashi, S.: Case Frame Compilation from the Web using High-Performance Computing, *Proc. 5th International Conference on Language Resources and Evaluation* (*LREC2006*), pp.1344–1347 (2006).

[2] Kaneda, K., Taura, K. and Yonezawa, A.: Virtual private grid: A command shell for utilizing hundreds of machines efficiently, *2nd IEEE/ACM International Symposium on Cluster Computing and the Grid* (*CCGrid 2002*) (2002).

[3] Kurohashi, S., Nakamura, T., Matsumoto, Y. and Nagao, M.: Improvements of Japanese morphological analyzer JUMAN, *The International Workshop on Sharable Natural Language*, pp.22–28 (1994).

[4] Kurohashi, S. and Nagao, M.: A syntactic analysis method of long Japanese sentences based on the detection of conjunctive structures, *Computational Linguistics*, Vol.20, No.4, pp.507–534 (1994).

[5] Shibata, T., Odani, M., Harashima, J., Oonishi, T. and Kurohashi, S.: SYNGRAPH: A Flexible Matching Method based on Synonymous Expression Extraction from an Ordinary Dictionary and a Web Corpus, *Proc. 3rd International Joint Conference on Natural Language Processing* (*IJCNLP2008*) (2008).

[6] Robertson, S.E., Walker, S., Hancock-Beaulieu, M., Gull, A. and Lau, M.: Okapi at TREC, *Text REtrieval Conference*, pp.21–30 (1992).

[7] Eguchi, K., Oyama, K., Ishida, E., Kando, N. and Kuriyama, K.: The Web retrieval task and its evaluation in the third NTCIR workshop, *Proc. 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (2003).

[8] Eguchi, K., Oyama, K., Aizawa, A. and Ishikawa, H.: Overview of WEB Task at the Fourth NTCIR Workshop, *Proc. 4th NTCIR Workshop on Research in Information Access Technologies Information Retrieval, Question Answering and Summarization* (2004).

[9] Jarvelin, K. and Kekalainen, J.: Cumulated gain-based evaluation of IR techniques, *ACM Trans. Information Systems*, Vol.20, pp.422–446 (2002).

[10] Fielding, R.T.: Architectural Styles and the Design of Network-based Software Architectures, PhD Thesis, University of California, Irvine (2000).

[11] Akamine, S., Kawahara, D., Kato, Y., Nakagawa, T., Leon-Suematsu, Y.I., Kawada, T., Inui, K., Kurohashi, S. and Kidawara, Y.: Organizing Information on the Web to Support User Judgments on Information Credibility, *Proc. 4th International Universal Communication Symposium* (*IUCS2010*), pp.122–129 (2010).

[12] Murakami, K., Nichols, E., Mizuno, J., Watanabe, Y., Masuda, S., Goto, H., Ohki, M., Sao, C., Matsuyoshi, S., Inui, K. and Matsumoto, Y.: Statement Map: Reducing Web Information Credibility Noise through Opinion Classification, *Proc. 4th Workshop on Analytics for Noisy Unstructured Text Data* (*AND 2010*), pp.59–66 (2010).

[13] Shibata, T., Banba, Y., Shinzato, K. and Kurohashi, S.: Web Information Organization using Keyword Distillation Based Clustering, *Proc. 2009 IEEE/WIC/ACM International Conference on Web Intelligence* (*WI-09*), pp.325–330 (2009).

[14] Harashima, J. and Kurohashi, S.: Summarizing Search Results using PLSI, *Proc. 2nd Workshop on NLPIX 2010*, pp.12–20 (2010).

[15] Turney, P.: Mining the Web for Synonyms: PMI-IR versus LSA on TOEFL, *Proc. 12th European Conference on Machine Learning* (*ECML-2001*), pp.491–502 (2001).

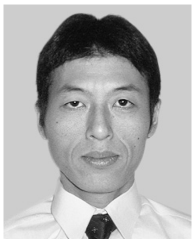
[16] Arms, W.Y., Aya, S., Dmitriev, P., Kot, B.J., Mitchell, R. and Walle, L.: Building a Research Library for the History of the Web, *Proc. Joint Conference on Digital Libraries*, pp.95–102 (2006).

[17] Popescu, A.-M., Etzioni, O. and Kautz, H.: Towards a Theory of Natural Language Interfaces to Databases, *Proc. 8th International Conference on Intelligent User Interfaces*, pp.149–157 (2003).

[18] Miyao, Y., Ohta, T., Masuda, K., Tsuruoka, Y., Yoshida, K., Ninomiya, T. and Tsujii, J.: Semantic retrieval for the accurate identification of relational concepts in massive textbases, *Proc. 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the ACL*, pp.1017–1024 (2006).

Keiji Shinzato received his B.E. from Tokyo Denki University, in 2002, and both M.S. and Ph.D. in Information Science from Japan Advanced Institute of Science and Technology in 2004, 2006, respectively. He is currently a senior scientist of Rakuten Institute of Technology. His research interests are natural language processing and information retrieval.

Tomohide Shibata received his B.E., M.S. and Ph.D. degrees in Information Science and Technology from the Tokyo University in 2002, 2004 and 2007, respectively. He is currently an assistant professor of Kyoto University, Japan. His research area is natural language processing.



Daisuke Kawahara received his B.S. and M.S. in Electronic Science and Engineering from Kyoto University in 1997 and 1999, respectively. He obtained his Ph.D. in Informatics from Kyoto University in 2005. He is currently an associate professor of the Graduate School of Informatics at Kyoto University. His research interests center on natural language processing, in particular knowledge acquisition and text understanding.

Sadao Kurohashi received his Ph.D. in Electrical Engineering from Kyoto University in 1994. He is currently a professor of the Graduate School of Informatics at Kyoto University. His research interests include natural language processing, knowledge acquisition/ representation, and information retrieval.