検索エンジン基盤 TSUBAKI

新里圭司　黒橋禎夫
{shinzato, kuro}@nlp.kuee.kyoto-u.ac.jp

2008 年 2 月 22 日

1　はじめに

本稿では，開放型検索エンジン基盤 TSUBAKI[*1]について述べる．開放型検索エンジン基盤 TSUBAKI とは，日本語ウェブページ約 1 億件[*2]を対象とした，研究用途に主眼をおいた検索エンジンである．API を公開しており[*3]，API を介して誰でも自由かつ無制限にその検索結果を取得することが可能である．また，ソースコードはオープンソースで公開する予定であり，独自に開発された検索エンジンコンポーネント (例えば，検索結果のランキングモジュールやリンク解析技術など) との融合を歓迎している．独自のコンポーネントを TSUBAKI に組み込みたい方は別途ご相談頂きたい．

本稿では，TSUBAKI の以下の点について述べる．

- 動作環境
- 検索対象とする文書集合
- インデックス
- 検索
- API の利用方法

2　動作環境

TSUBAKI は国立情報学研究所 西千葉分館にて運用されている．表 1，2 に，西千葉分館の計算機環境を示す．ここでは，128 ある CPU コア全てが検索用に用いられており，一部の CPU コアはインデックス作成用などの処理にも併用される．

3　検索対象となる文書集合

TSUBAKI が検索対象としている文書集合は，2007 年 5 月から 7 月にかけて情報通信研究機構 知識処理グループにてクロールされた日本語ページ 100,380,002 件である．クロールには，東京大学田浦研究室にて開発

[*1] http://tsubaki.ixnlp.nii.ac.jp/index.cgi，科学技術研究費特定領域「情報爆発時代に向けた新しい IT 基盤技術の研究」(2006-2010; 領域代表 喜連川優) において開発・運用中.
[*2] 2007 年 10 月現在.
[*3] http://tsubaki.ixnlp.nii.ac.jp/tsubaki/api.html

表 1　TSUBAKI 系計算機環境 (計算サーバー)

	HP 社製ブレードサーバー (32 台)	APPRO 社製ブレードサーバー (16 台)	合計
CPU	64CPU コア (2CPU コア× 32)	64CPU コア (4CPU コア× 16)	128CPU コア
メモリ	192GB(6GB × 32)	192GB(12GB × 16)	384GB
ローカルディスク	19.2TB(0.6TB × 32)	16.0TB (1TB × 16)	35.2TB

表 2　TSUBAKI 系計算機環境 (ファイルサーバー)

ホスト名	容量
nlpcf.ixnlp.nii.ac.jp	2.0TB
nlpcf2.ixnlp.nii.ac.jp	10.5TB
nlpcf3.ixnlp.nii.ac.jp	53.4TB
nlpcf4.ixnlp.nii.ac.jp	53.4TB
合計	119.3TB

が進められている Shim-Crawler[*4]を利用した．

クロールされたウェブページは，日本語ウェブページ判定処理，日本語文区切り判定処理が施され，ウェブ標準フォーマット [2] により管理される．ウェブ標準フォーマットとは，ウェブページを解析した結果の共有を目的に，我々が提案した XML 形式のフォーマットであり，ウェブページを対象とした研究を行う上で頻繁に利用される情報，例えば，ウェブページのタイトル，URL，ページに含まれる日本語文などの情報が埋め込まれる．本節ではウェブ標準フォーマット及び，ウェブページから標準フォーマットへの変換方法について述べる．

3.1　ウェブ標準フォーマット

ウェブページを共有するための管理方式として，XML タグによりページの情報を埋め込むことのできるウェブ標準フォーマットを設計した．表 3 に標準フォーマットに埋め込まれる情報を示す．表からわかるように，このフォーマットにはウェブページのタイトルや URL，リンク情報などのメタ情報に加え，ウェブページを扱うシステムで多くの場合利用される文情報，さらにオプションとして，文を既存の自然言語処理ツールにより解析した結果を埋め込むことが可能である．

図 1(a) にオリジナルのウェブページを，同図 (b) に標準フォーマットに変換された文書（KNP [5] の解析結果付き）をそれぞれ示す．標準フォーマット化されたウェブページは大きく分けると，<Header>タグで囲まれるヘッダー部と<Text>タグで囲まれるテキスト部から構成される．ヘッダー部は，対応するウェブページのタイトル (<Title>)，ウェブページへのインリンク情報 (<InLink>) および，ウェブページからのアウトリンク情報 (<OutLink>) などのメタ情報を含んでいる[*5]．例えば，図 2 に示した ID 300 のウェブページを標準フォーマット化すると，図??に示すヘッダー部が得られる．

その一方で，テキスト部には，対応するウェブページから抽出された日本語文および，日本語文を自然言語処理ツールを使って解析した結果が含まれる．テキスト部の例を図 4 に示す．<S>タグで囲まれた範囲が，オリジナルのウェブページから抽出された日本語文 1 文に対応しており，その子要素である<RawString>タグにより実際に抽出された日本語文が，<Annotation>タグにより抽出された日本語文を自然言語処理ツールで

[*4] http://www.logos.t.u-tokyo.ac.jp/crawler/
[*5] インリンク，アウトリンクについては現在処理中である．

表 3　標準フォーマットに明示されている情報

タグ名	内容
StandardFormat	標準フォーマットのルートタグ． **属性**:　Url:　元となるウェブページの URL OriginalEncoding:　元となるウェブページの文字コード Time:　ページが取得された日時（「yyyy-mm-dd hh:mm:ss」形式） **子要素**:　Header，Text
Header	ヘッダー要素を表すタグ． **属性**:　なし **子要素**:　Title，InLinks，OutLinks
Title	元となるページのタイトルを表すタグ． **属性**:　Offset:　ファイル先頭からのオフセット Length:　タイトルの長さ（バイト長） Id:　文 ID **子要素**:　RawString，Annotation
InLinks	元となるページへのインリンクの集合を表すタグ． **属性**:　なし **子要素**:　InLink
InLink	元となるページへのインリンクを表すタグ． **属性**:　なし **子要素**:　RawString，Annotation，DocIDs
OutLinks	元となるページからのアウトリンクの集合を表すタグ． **属性**:　なし **子要素**:　OutLink
OutLink	元となるページからのアウトリンクを表すタグ． **属性**:　なし **子要素**:　RawString，Annotation，DocIDs
DocIDs	文書 ID の集合を表すタグ． **属性**:　なし **子要素**:　DocID
DocID	文書 ID を表すタグ． **属性**:　なし **子要素**:　文書 ID を表す数字列
Text	元となるページの本文の内容を表すタグ． **属性**:　Type:　ウェブページのタイプ（通常のページ，ブログのエントリなど） **子要素**:　S
S	ページに含まれる一文を表すタグ． **属性**:　Offset:　ファイル先頭からのオフセット Length:　タイトルの長さ（バイト長） Id:　文 ID **子要素**:　RawString，Annotation
RawString	一文として抽出された文字列を表すタグ． **属性**:　なし **子要素**:　一文として抽出された文字列
Annotation	一文として抽出された文字列の解析結果を表すタグ． **属性**:　Scheme:　解析に用いたツール名 (e.x., Juman，Knp など) **子要素**:　ツールの解析結果

解析した結果がそれぞれ囲まれている．また，同一の日本語文を複数の自然言語処理ツールで解析する場合を考慮して，`<Annotation>`タグの `Scheme` 属性の値を異なる値 (基本的にはツール名) にすることで，複数の解析結果を同一の標準フォーマットに埋め込めるようになっている．

3.2　標準フォーマットへの変換処理

ウェブページから標準フォーマットデータを生成するためのステップを以下に示す．

(a) ウェブページ　　(b) 標準フォーマット(構文解析結果付き)

図 1　標準フォーマット化されたウェブページの例

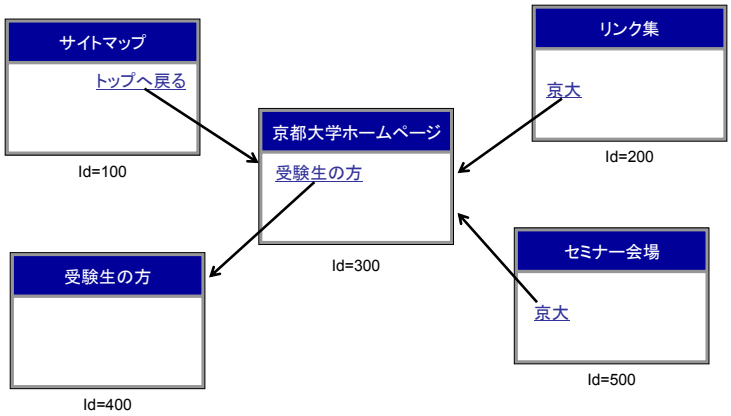


図 2　インリンク，アウトリンクの例

Step 1: 日本語ウェブページかどうかの判定（オプション）
Step 2: ウェブページの utf8 化
Step 3: ウェブページからの日本語文抽出
Step 4: 抽出された日本語文の解析（オプション）
Step 5: ウェブ標準フォーマット形式に変換

ここでは各ステップについて述べる．

■Step 1: **日本語ウェブページかどうかの判定（オプション）**　文字コードおよび言語情報を手がかりにウェブページが日本語ページかどうかの判定を行う．

(i) **文字コードを用いた日本語ページ判定**:　ウェブページ中の charset 属性, または perl の `Encode::guess_encoding()`

```
<Header>
  <Title>
    <RawString>京都大学ホームページ</RawString>
  </Title>
  <OutLinks>
    <OutLink>
      <RawString>受験生の方へ</RawString>
      <DocIDs>
        <DocID>400</DocID>
      </DocIDs>
    </OutLink>
  </OutLinks>
  <InLinks>
    <InLink>
      <RawString>京大</RawString>
      <DocIDs>
        <DocID>300</DocID>
        <DocID>500</DocID>
      </DocIDs>
    </InLink>
    <InLink>
      <RawString>トップへ戻る</RawString>
      <DocIDs>
        <DocID>300</DocID>
      </DocIDs>
    </InLink>
  </InLinks>
</Header>
```

図 3　ヘッダー部の例

関数を用いてページの文字コードを調べる．文字コードが euc-jp，x-euc-jp，iso-2022-jp，shift jis，windows-932，x-sjis，shiftjp，utf-8 であれば，そのページを日本語ページの候補とする．

(ii) **言語情報を用いた日本語ページ判定**: (i) では，より多くの日本語ページを取得するために，utf8 コードでエンコードされているページも日本語ページとして見なしている．しかし，utf8 コードは日本語以外の言語でも用いられるため，日本語以外のページが含まれている可能性がある．ここでは言語情報を用いて utf8 でエンコードされたページのうち，日本語以外のページを排除する．具体的には，日本語の助詞（「が」「を」「に」等）に注目し，助詞が一定以上の割合で含まれていないページは日本語ページでないと判定する．

■Step 2: **ウェブページの** utf8 **化**: Step 1 (i) の操作によりウェブページの文字コードを調べ，ウェブページの文字コードを utf8 に変換する．この時，ページ中に charset 属性が指定されている場合は，charset 属性の値を utf8 に変更する．

■Step 3: **ウェブページからの日本語文抽出**　日本語文の境界を認識し，ウェブページの本文を文単位に区切る．文境界認識のための手がかりとしては HTML タグおよび句点を利用する．文認識のための手かがりとして用いる HTML タグとして，
と<p>を利用する．また<pre>タグ中の改行は，そのまま改行として扱う．

■Step 4: **抽出された日本語文の解析（オプション）**　抽出された日本語文を既存の言語処理ツールを用いて解析する．

```
<Text Type="default">
<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="KNP">
    <![CDATA[* 1D <文頭><サ変><人名><助詞><連体修飾><体言><係:ノ格><区切:0-4><RID:1056>
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>
 ... 中略...
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>
    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="KNP">
    <![CDATA[* 1D <文頭><助詞><連体修飾><体言><係:ノ格><区切:0-4><RID:1056>
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>
 ... 中略...
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>
EOS]]>
  </Annotation>
</S>
... 中略...
</Text>
```

図 4　テキスト部の例 (KNP による解析結果有)

■Step 5: **ウェブ標準フォーマット形式に変換**　抽出された日本語文，日本語文の解析結果（オプション）を標準フォーマット形式で保存する．文書取得日時，URL などウェブページのメタ情報は，クローラより得られる情報を利用する．

ウェブ標準フォーマットへの変換手順を，先述のクロールされた 2 億ページに対して適用し，日本語ウェブページの抽出及び，抽出されたページの標準フォーマット化 (KNP の解析結果付き) を行った．我々が今回用いた標準フォーマット変換ツールは以下のアドレスよりダウンロード可能である．

http://nlp.kuee.kyoto-u.ac.jp/ skeiji/html2sf.tgz

標準フォーマット変換に用いた計算機環境は，Intel CPU Xeon 3.0GHz × 4，メモリ 4GB のスペックを持つ計算機 162 台であり，GXP2[4] を用いて並列に変換処理を行った．

その結果，約 1 億件の標準フォーマットデータが得られた．標準フォーマットデータの生成には約 2 週間かかった．これはつまり，残りの 1 億ページについては，日本語ページとして見なされなかったことを意味している．このウェブ標準フォーマット化されたデータは，後述する TSUBAKI API を利用することで，誰でも容易に取得することが可能である．また，InTrigger ユーザーであれば，これらのデータは InTrigger(chiba) 上にコピー済みであるため，API を用いなくても直接利用可能である．

表 4　オリジナルのウェブページと標準フォーマットデータのファイルサイズ (約 1 億ページ，gzip 圧縮時)

ファイルの種別	サイズ [TB]
オリジナルのウェブページ (utf8 変換前)	0.6
標準フォーマット変換済データ	3.1

表 4 に標準フォーマット化されたデータ（約 1 億件）および，対応するウェブページのファイルサイズを示す．

3.3　InTrigger での標準フォーマットデータの利用方法

前節で生成した一億件の KNP 解析結果付き標準フォーマットデータは InTrigger (chiba) のファイルサーバー上にコピー済みである．そのため，InTrigger(chiba) 上のクラスタマシンを使って標準フォーマットデータを直接処理することが可能である．標準フォーマットデータは，`/data/[1,2]/skeiji` 以下にある．

```
skeiji@chiba100:~$ ls /data/1/skeiji/sfs
x000  x004  x008  x012  x016  x020  x024  x028  x032  x036  x040  x044  x048
x001  x005  x009  x013  x017  x021  x025  x029  x033  x037  x041  x045  x049
x002  x006  x010  x014  x018  x022  x026  x030  x034  x038  x042  x046
x003  x007  x011  x015  x019  x023  x027  x031  x035  x039  x043  x047
skeiji@chiba100:~$ ls /data/2/skeiji/sfs
x050  x054  x058  x062  x066  x070  x074  x078  x082  x086  x090  x094  x098
x051  x055  x059  x063  x067  x071  x075  x079  x083  x087  x091  x095  x099
x052  x056  x060  x064  x068  x072  x076  x080  x084  x088  x092  x096  x100
x053  x057  x061  x065  x069  x073  x077  x081  x085  x089  x093  x097
skeiji@chiba100:~$
```

`/data/1/skeiji` および`/data/2/skeiji` 上にそれぞれ 50 個と 51 個のディレクトリがあり，各ディレクトリ内には 100 個の `tgz` ファイルが置かれている．

```
skeiji@chiba100:~$
skeiji@chiba100:~$ ls /data/1/skeiji/sfs/x000
x00000.tar.gz  x00017.tar.gz  x00034.tar.gz  x00051.tar.gz  x00068.tar.gz  x00085.tar.gz
x00001.tar.gz  x00018.tar.gz  x00035.tar.gz  x00052.tar.gz  x00069.tar.gz  x00086.tar.gz
x00002.tar.gz  x00019.tar.gz  x00036.tar.gz  x00053.tar.gz  x00070.tar.gz  x00087.tar.gz
... 中略 ...
x00014.tar.gz  x00031.tar.gz  x00048.tar.gz  x00065.tar.gz  x00082.tar.gz  x00099.tar.gz
x00015.tar.gz  x00032.tar.gz  x00049.tar.gz  x00066.tar.gz  x00083.tar.gz
x00016.tar.gz  x00033.tar.gz  x00050.tar.gz  x00067.tar.gz  x00084.tar.gz
skeiji@chiba100:~$
```

各 `tgz` ファイルは，10,000 個の標準フォーマットデータを圧縮したものになっている．

```
skeiji@chiba100:~$ cp /data/1/skeiji/sfs/x000/x00000.tar.gz ~/
skeiji@chiba100:~$ tar xzf x00000.tar.gz
skeiji@chiba100:~$ ls x00000
000000000.xml.gz  000002000.xml.gz  000004000.xml.gz  000006000.xml.gz  000008000.xml.gz
000000001.xml.gz  000002001.xml.gz  000004001.xml.gz  000006001.xml.gz  000008001.xml.gz
... 中略 ...
000001999.xml.gz  000003999.xml.gz  000005999.xml.gz  000007999.xml.gz  000009999.xml.gz
000001998.xml.gz  000003998.xml.gz  000005998.xml.gz  000007998.xml.gz  000009998.xml.gz
skeiji@chiba100:~$
```

シングルプロセスにて標準フォーマットデータを処理する際は，ファイルサーバー上のファイルを直接処理しても問題はない．しかしながら，複数台のマシンを使ってファイルサーバー上のデータを並列処理しようとする場合は，ファイルサーバーに対して高負荷がかかるため問題となる．そこで，複数台のクラスタを使って並列処理を行えるように，標準フォーマットデータは chiba[100-158] までの各ローカルディスク上に分散してコピーされている．具体的には，各マシンの/data/local/skeiji/sfs 以下に 3，4 個ずつ tgz ファイルが置かれている．標準フォーマットデータを並列処理する際は，ファイルサーバーへ高負荷を与えないようにするために，chiba[100-158] のローカルディスク上に分散されたファイルを，scp などを用いてコピーしてから処理して頂きたい．

4 インデックス

本節では，TSUBAKI が検索時に利用するインデックスについて述べる．

4.1 インデックスの単位

TSUBAKI では，インデックスの単位として以下のものを採用している．

- 単語（自立語，機能語全て）
- 自立語同士の係り受け関係
- 同義表現（単語，句）
- 同義表現同士の係り受け関係

単語および自立語同士の係り受け関係については KNP の解析結果から, 同義表現, 同義表現同士の係り受け関係については SynGraph [1] の解析結果から取得している．これらの解析結果はあらかじめ標準フォーマットに埋め込まれているものを利用するため，インデックス作成のために新たにウェブページを解析することはしない．

以下では各インデックスについて述べる．

4.1.1 単語インデックス

TSUBAKI では，KNP の解析結果から代表表記化された単語およびその読みを単語インデックスとして抽出する．「子ども服をせんたくする」という文を KNP で解析した結果を図 5 に示す．この結果からは以下の単語がインデックスとして抽出される．

子供／こども，服／ふく，を／を，洗濯／せんたく，選択／せんたく，する／する

```
* 1D <BGH:子供/こども ＋ 服/ふく><文頭><ヲ><助詞><体言><係:ヲ格><区切:0-0><RID:1118><格要素><連用要素>
＋ 1D <SM-主体><SM-人><BGH:子供/こども><文節内><係:文節内><文頭><体言><名詞項候補><先行詞候補>
子ども こども 子ども 名詞 6 普通名詞 1 * 0 * 0 "カテゴリ:人 代表表記:子供/こども" <カテゴリ:人><代表表記:子供/こども><文頭><かな漢字><名詞相当語><自立><意味有><タグ単位始><文節始>
＋ 2D <BGH:子供/こども ＋ 服/ふく><ヲ><助詞><体言><係:ヲ格><区切:0-0><RID:1118><格要素><連用要素><名詞項候補><先行詞候補>
服 ふく 服 名詞 6 普通名詞 1 * 0 * 0 "漢字読み:音 カテゴリ:人工物-衣類 代表表記:服/ふく" <漢字読み:音><カテゴリ:人工物-衣類><代表表記:服/ふく><漢字><かな漢字><名詞相当語><自立><複合←><意味有><タグ単位始><文節主辞>
を を を 助詞 9 格助詞 1 * 0 * 0 NIL <かな漢字><ひらがな><付属>
* -1D <BGH:洗濯/せんたく ＋ する/する><文末><サ変><サ変スル><用言:動><レベル:C><区切:5-5><ID:（文末）><RID:112><提題受:30><主節>
＋ -1D <BGH:洗濯/せんたく ＋ する/する><文末><サ変スル><用言:動><レベル:C><区切:5-5><ID:（文末）><RID:112><提題受:30><主節:><サ変><主題格:一人称優位>
せんたく せんたく せんたく 名詞 6 サ変名詞 2 * 0 * 0 "カテゴリ:抽象物 ドメイン:家庭・暮らし 代表表記:洗濯/せんたく" <カテゴリ:抽象物><ドメイン:家庭・暮らし><代表表記:洗濯/せんたく><品曖><ALT-せんたく-せんたく-せんたく-6-2-0-0-"カテゴリ:抽象物 代表表記:選択/せんたく"><品曖-サ変名詞><品曖-その他><原形曖昧><かな漢字><ひらがな><名詞相当語><サ変><サ変動詞><自立><意味有><タグ単位始><文節始><文節主辞>
する する する 動詞 2 * 0 サ変動詞 16 基本形 2 "付属動詞候補（基本） 代表表記:する/する" <付属動詞候補（基本）><代表表記:する/する><文末><表現文末><とタ系連用テ形複合辞><かな漢字><ひらがな><連体修飾><活用語><付属>
EOS
```

図 5 「子ども服をせんたくする」を KNP で構文解析した結果

図 6 「子ども服をせんたくする」を KNP で解析して得られる係り受け関係（代表表記化済み，下線が引かれている単語は自立語）

KNP の解析結果を用いることで，「子ども」のような表記揺れを持つ単語はその語の代表的な表記（例えば，「子供」）に変換される．これにより，検索クエリまたは文書中に出現する単語の表記揺れに起因した検索漏れを軽減する効果が期待できる．また，「せんたく」のように複数の代表表記（「洗濯」，「選択」）が考えられる単語については，その全ての表記が文書中に出現したものと見なす．この時，各単語の出現頻度は，代表表記候補の個数で割ったものになる．上の例文の場合，「せんたく」は，「洗濯」および「選択」が出現したものと見なされ，各単語の出現頻度は 0.5 とカウントされる．

単語インデックスは，索引語の他に以下の情報を持っている．

- 単語自身
- 単語の文書頻度
- 単語が出現した文書 ID
- 各文書中での単語の出現頻度
- 単語の出現した文 ID
- 単語の出現した位置

フレーズ検索や，複数の単語が同一文中で共起していることを制約にした検索を実現するため，単語の出現位置および単語の出現した文 ID（標準フォーマットから抽出可能）をインデックスに登録している．なお，単語の出現位置は文書スコアの計算時にも利用される．TSUBAKI では上記の情報をバイナリ化した独自のフォーマットで管理している．バイナリ化されたインデックスデータの構造を図 7 に示す．

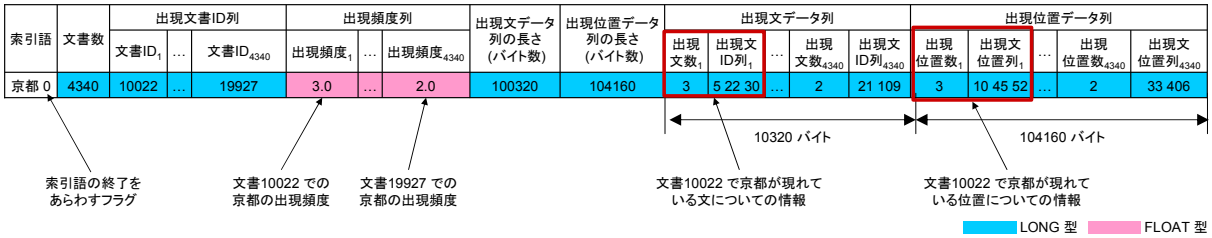


図 7　インデックスのフォーマット

4.1.2　係り受け関係インデックス

TSUBAKI では，単語に加え，係り受け関係もインデックスとして抽出する．具体的には，自立語同士の係り受け関係を係り受け関係インデックスとして抽出する．図 5 に挙げた KNP の解析結果から得られる係り受け関係を図 6 に示す．これより，「子ども服をせんたくする」からは，

子供／こども→服／ふく，服／ふく→洗濯／せんたく，服／ふく→選択／せんたく

が係り受けインデックスとして抽出される．単語の場合と同様に，係り先または係り元に複数の代表表記候補がある場合は，その全ての組み合わせをインデックスとして抽出し，各インデックスの出現頻度は，候補数で割ったものとしている．

係り受けインデックスに含まれる情報は以下の通りである．

- 係り受け自身
- 係り受けの文書頻度
- 係り受けが出現した文書 ID
- 各文書中での係り受けの出現頻度

単語インデックスとは異なり，出現文に関する情報および出現位置に関する情報は含まれないことに注意されたい．

4.1.3　同義表現インデックス

TSUBAKI では，代表表記による表記揺れを吸収した検索に加え，単語または句の同義表現についてもインデックスを作成することで，より柔軟な検索を実現している．例えば，クエリとして「乳児の虫歯を予防する」が与えられた場合，「乳児」が「赤ちゃん」と同義であること，「予防」が「防ぐ」と同義であることを同時に考慮した検索が行われる．その結果として，「赤ちゃんの虫歯を防ぐには」といったクエリとは異なる表現が含まれる文書であっても検索結果として得ることが可能となる．

クエリおよび文書中の単語・句の同義表現を獲得するにあたり，SynGraph の解析結果を利用する．SynGraph とは，国語辞典およびウェブページからあらかじめ構築しておいた同義表現データベースを利用して，入力文中に含まれる単語・句に対して，それらと同義である単語・句を出力するツールである．図 8 に「赤ちゃんの虫歯を予防する」を SynGraph により解析した結果を，図 9 に同義表現間の係り受け関係を示す．TSUBAKI では，SynGraph の解析結果に含まれる基本ノードおよび Syn ノードと呼ばれるデータ[*6]を，各

[*6] 基本ノードおよび Syn ノードについては，該当論文 [1] を参照されたい．

```
+ 1D <SM-主体><SM-人><BGH:乳児/にゅうじ><文頭><助詞><連体修飾 ><体言><係:ノ格><区切:0-4><RID:1069><名詞項補> <先行詞候補><係チ:非用言格解析――用言&&文節内:T解析格-ヲ>
乳児 にゅうじ 乳児 名詞 6 普通名詞 1 * 0 * 0 "カテゴリ:人 ドメイン:健康・医学:家庭・暮らし 代表表記:乳児/にゅうじ" <カテゴリ:人><ドメイン:健康・医学:家庭・暮らし><代表表記:乳児/にゅうじ><文頭> <漢字><かな漢字><名詞相当語><自立><意味有><タグ単位始> <文節始><文節主辞>
の の の 助詞 9 接続助詞 3 * 0 * 0 NIL <かな漢字><ひらがな><付属>
!! 0 1D <見出し:乳児の>
! 0 <SYNID:乳児/にゅうじ><スコア:1>
! 0 <SYNID:s121:乳飲み子/ちのみご><スコア:0.99>
! 0 <SYNID:s2894:児童/じどう><スコア:0.693><上位語><下位語数:21>
! 0 <SYNID:s14116:子><スコア:0.693><上位語><下位語数:16>
! 0 <SYNID:s2669:子女/しじょ><スコア:0.693><上位語><下位語数:21>
! 0 <SYNID:s2787:小児/しょうに><スコア:0.693><上位語><下位語数:21>
+ 2D <BGH:虫歯/むしば><ヲ><助詞><体言><係:ヲ格><区切:0- 0><RID:1118><格要素><連用要素><名詞項候補><先行詞候補> <解析格:ヲ>
虫歯 むしば 虫歯 名詞 6 普通名詞 1 * 0 * 0 "カテゴリ:動物-部位 ドメイン:健康・医学代表表記:虫歯/むしば" <カテゴリ:動物-部位><ドメイン:健康・医学><代表表記:虫歯/むしば><漢字><かな漢字><名詞相当語><自立><意味有><タグ単位始><文節始><文節主辞 >
を を を 助詞 9 格助詞 1 * 0 * 0 NIL <かな漢字><ひらがな><付属>
!! 1 2D <見出し:虫歯を><格解析結果:ヲ格>
! 1 <SYNID:虫歯/むしば><スコア:1>
! 1 <SYNID:s15141:虫歯/むしば><スコア:0.99>
! 1 <SYNID:s15137:歯/は><スコア:0.693><上位語><下位語数:9>
+ -1D <BGH:予防/よぼう ＋ する/する><文末><サ変スル><用言:動><レベル:C> <区切:5-5><ID:（文末）><RID:112><提題受:30><主節><　サ変><格要素-ガ:＃一人称優位><格要素-ヲ:虫歯><格要素-ニ:NIL><格要素-ト:　NIL><格要素-デ:NIL><格要素-カラ:NIL><格要素-ヨリ:NIL><格要素-マデ:　NIL><格要素-時間:NIL><格要素-外の関係:NIL><格要素-ノ:NIL><格要素-ニヨル:NIL><格要素-ヲツウジル:NIL><格要素-ニタイスル:NIL><格要素-トスル:NIL> <格要素-ニカンスル:NIL><格要素-ニクワエル:NIL><格要素-ヲハジメル:NIL><格要素-ニオク:NIL><格要素-ヲノゾク:NIL><格フレーム-ガ-主体準><格フレーム-ガ-主体ｏｒ主体準> <主題格:一人称優位><格関係 1:ヲ:虫歯><格解析結果:予防/よぼう:動 1:ガ/U/-/-/-/-; ヲ/C/虫歯 /1/0/?; ニ/U/-/-/-/-; ト/U/-/-/-/-; デ/U/-/-/-/-; カラ/U/-/-/-/-; ヨリ/U/-/-/-/-; マデ /U/-/-/-/-; 時間/U/-/-/-/-; 外の関係/U/-/-/-/-; ノ/U/-/-/-/-; ニヨル/U/-/-/-/-; ヲツウジル /U/-/-/-/-; ニタイスル/U/-/-/-/-; トスル/U/-/-/-/-; ニカンスル/U/-/-/-/-; ニクワエル/U/-/-/-/ -; ヲハジメル/U/-/-/-/-; ニオク/U/-/-/-/-; ヲノゾク/U/-/-/-/->
予防 よぼう 予防 名詞 6 サ変名詞 2 * 0 * 0 "カテゴリ:抽象物 ドメイン:健康・医学 代表表記:予防/よぼう" <カテゴリ:抽象物><ドメイン:健康・医学><代表表記:予防/よぼう><漢字><かな漢字 ><名詞相当語><サ変><サ変動詞><自立><意味有><タグ単位始 ><文節始><文節主辞>
する する する 動詞 2 * 0 サ変動詞 16 基本形 2 "付属動詞候補（基本） 代表表記:する/する" <付属動詞候補（基本）><代表表記:する/する><文末><表現文末><とタ系連用テ形複合辞 ><かな漢字><ひらがな><活用語><付属>
!! 2 -1D <見出し:予防する>
! 2 <SYNID:予防/よぼう><スコア:1>
! 2 <SYNID:s17557:予防/よぼう><スコア:0.99>
! 2 <SYNID:s4234:防ぐ/ふせぐ><スコア:0.693><上位語><下位語数:3>
! 2 <SYNID:s35:回避/かいひ><スコア:0.693><上位語><下位語数:3>
EOS
```

図 8　「乳児の虫歯を予防する」を SynGraph で解析した結果

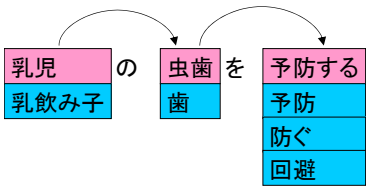


図 9　「乳児の虫歯を予防する」を SynGraph 構造に変換した例（赤は基本ノードを，青は Syn ノードを意味する）

語・句の同義表現インデックスとして抽出する．例えば，図 8 からは，

<SYNID:乳児/にゅうじ>, <SYNID:s121:乳飲み子/ちのみご>, <SYNID:虫歯/むしば>, <SYNID:s15141:虫歯/むしば>, <SYNID:s15137:歯/は>, <SYNID:予防/よぼう>, <SYNID:s17557:予防/よぼう>, <SYNID:s4234:防ぐ/ふせぐ>, <SYNID:s35:回避/かいひ>

がインデックスとして抽出される．同義表現インデックスでは，フレーズ検索を想定していないため，単語インデックスのように全ての語をインデックスとして抽出しない．

また，各同義表現インデックスの出現頻度は，SynGraph により与えられるスコアを利用している．例えば，基本ノード<SYNID:乳児/にゅうじ> については 1 が，Syn ノード<SYNID:乳児/にゅうじ>については 0.99 が，同じく Syn ノード<SYNID:予防/よぼう>については 0.693 がそれぞれ各インデックスの出現頻度として登録される．

同義表現インデックスは以下の情報を含んでいる．

- 同義表現自身
- 同義表現の文書頻度
- 同義表現が出現した文書 ID
- 各文書中での同義表現の出現頻度
- 同義表現の出現した位置

これらの情報は，単語・係り受けインデックス同様，バイナリ化され保持される．

4.1.4 同義表現同士の係り受けインデックス

TSUBAKI では，同義表現についても，同義表現同士の係り受け関係をインデックスとして保持する．具体的には，基本ノード，Syn ノード間の係り受け関係をインデックスとして抽出する．例えば，図 8 に示した SynGraph の解析結果からは以下の係り受け関係が抽出される．

<SYNID:乳児/にゅうじ>→<SYNID:虫歯/むしば>, <SYNID:乳児/にゅうじ>→<SYNID:s15141:虫歯/むしば>, . . . 中略 . . . <SYNID:s15137:歯/は>→<SYNID:s35:回避/かいひ>

同義表現インデックス同様，出現頻度は，SynGraph の解析結果より得られるスコアを利用しており，具体的には，係り先と係り元の単語・句に対する同義表現の持つスコア同士の積を，係り受けインデックスの出現頻度としている．同義表現の係り受けインデックスについては，インデックスデータのサイズを小さくするために，他のインデックスとは異なり，文書頻度が 10 未満の索引表現についてはインデックスとして登録していない．

最後に，同義表現同士の係り受けインデックスに登録されている情報を以下に示す．

- 同義表現の係り受け関係自身
- 同義表現の係り受け関係の文書頻度
- 同義表現の係り受け関係が出現した文書 ID
- 各文書中での同義表現の係り受け関係の出現頻度
- 同義表現の係り受け関係が出現した位置

4.2 インデックスデータのサイズ

表 5 に各インデックスデータに含まれる情報，足切りの有無，ファイルサイズを示す．表より同義表現の係り受けインデックスのファイルサイズが大きいことがわかる．また，係り受けインデックスに比べ，単語インデックスの方がファイルサイズが大きくなっているが，これは，単語インデックスが全単語の全出現位置をインデックスとして登録しているのに対し，係り受け関係インデックスの方は，自立語同士の係り受け関係に限定，さらに位置情報を保持していないためである．

表 5　TSUBAKI で用いるインデックスデータ

インデックスの種別	索引表現自身	文書頻度	出現文書情報	出現文情報	出現位置情報	文書頻度による足切り	サイズ [TB]
単語	○	○	○	○	○	×	1.17
係り受け	○	○	○	×	×	×	0.89
同義表現	○	○	○	×	○	×	1.84
同義表現の係り受け	○	○	○	×	○	○	4.81

表 6　TSUBAKI で指定可能な検索キーワード制約

制約名	タグ名	説明	例
AND 制約	~AND	検索キーワード中の単語（自立語）が全て含まれていなければならない	京都大学~AND
OR 制約	~OR	検索キーワード中のいずれかの単語（自立語）が含まれている	京都大学~OR
フレーズ制約	タグなし．“”’ で囲む	既存の検索エンジンで使われているフレーズ検索に相当する	”京都大学”
係り受け制約	~FD	検索キーワード中の係り受け関係が全て含まれていなくてはならない	京都大学~FD
近接制約（文）	~NS	検索キーワード中の単語（自立語）が，キーワード中の出現順で，N 文以内に現れている	京都大学~5S
近接制約（単語）	~NW	検索キーワード中の単語（自立語）が，キーワード中の出現順で，N 語以内に現れている	京都大学~20W

5　検索

5.1　指定可能な検索制約

TSUBAKI では，入力として与えられた複数個の検索キーワード間に，AND/OR の論理条件を適用して検索することが可能である（一般に言われている AND 検索・OR 検索に相当する）が，検索条件に加え，検索キーワードごとに**制約**を指定することも可能である．制約は，検索キーワードの末尾に「~制約を表す文字列」を指定して表現する．表 6 に TSUBAKI で指定可能な検索キーワード制約を示す．

5.2　文書のスコアリング

TSUBAKI では，ユーザより与えられた検索条件および検索クエリに従って，条件に一致する文書を検索する．そして，検索して得られた文書群に対して，検索クエリとの適合度を表すスコアを計算する．検索クエリ Q（子どもの体力低下）から抽出される単語インデックスを Q_{word}（子供，体力，低下），係り受けインデックス Q_{dpnd}（子供→体力，体力→低下）とした時，検索クエリ Q に対する文書 d のスコア $score_{rel}(Q,d)$ を以下の式により計算する．

$$score_{rel}(Q,d) = rel_{word}(Q_{word},d) + rel_{dpnd}(Q_{dpnd},d)$$

ここで，$rel_{word}(Q_{word},d)$ は Q の単語インデックスより計算されるスコア，$rel_{dpnd}(Q_{dpnd},d)$ は Q の係り受けインデックスより計算されるスコアをそれぞれ意味する．TSUBAKI では，$rel_{word}(Q_{word},d)$ および $rel_{dpnd}(Q_{dpnd},d)$ に対して重み付けを行っていない．これは，単語に比べ係り受け関係の出現頻度が相対的に低いことを利用し，出現頻度が低い表現程高いスコアを得られるよう $rel_{dpnd}(Q_{dpnd},d)$ を設計しているためである．そのため，$rel_{dpnd}(Q_{dpnd},d)$ に対して特別なボーナスを与えることをしなくとも，Q 中の係り受

け関係を含んでいる文書に対して自然な形で高いスコアを与えることができている．

$rel_{word}(Q_{word}, d)$ は OKAPI BM25[6] を基にした以下の式により求められる．

$$rel_{word}(Q_{word}, d) = \sum_{q \in Q_{word}} w \times \frac{(k_1+1)fq}{K+fq} \times \frac{(k_3+1)qfq}{k_3+qfq}$$

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

ここで，fq は表現 q の文書 d 中での出現頻度，qfq は q の検索クエリ中での出現頻度，N は検索対象としている全文書数 (TSUBAKI では 1.0×10^8)，n は q を含む文書数，l は文書 d の文書長 (TSUBAKI では文書中の単語数)，l_{ave} は平均文書長を表す．また，k_1,k_3,b は *BM25* のパラメータであり，TSUBAKI では，$k_1 = 2$, $k_3 = 0$, $b = 0.75$ を用いている．

一方，$rel_{dpnd}(Q_{dpnd}, d)$ は以下の式により求められる．

$$rel_{dpnd}(Q_{dpnd}, d) = \sum_{q \in Q_{dpnd}} f(q, d)$$

$$f(q,d) = \begin{cases} \dfrac{3 \times fq}{K+fq} \times \log \dfrac{N-n+0.5}{n+0.5} & q \text{ が } d \text{ に含まれている場合} \\ \dfrac{3 \times w(q)}{K+fq} \times \log \dfrac{N-n+0.5}{n+0.5} & otherwise \end{cases}$$

ここで $w(q)$ は

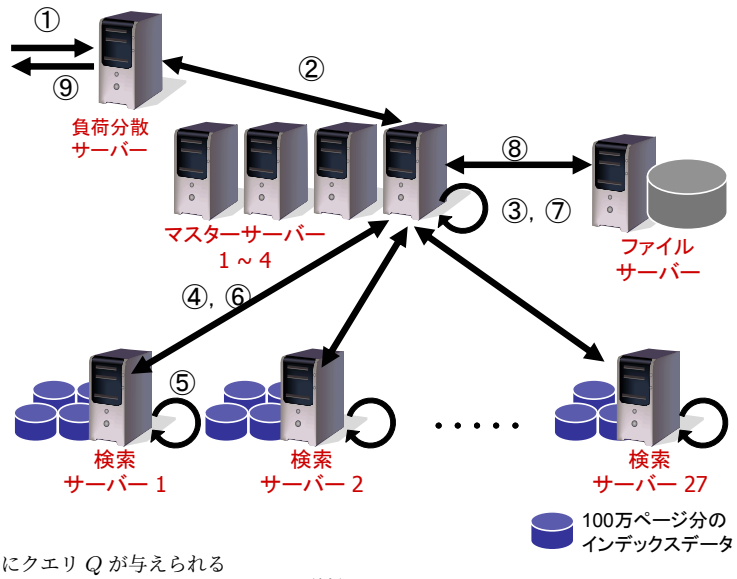
$$w(q) = \begin{cases} \dfrac{D - \text{min\_dist}(l(q), r(q))}{D} & \text{min\_dist}(l(q), r(q)) < D \\ 0 & otherwise \end{cases}$$

で表される関数である．$\text{min\_dist}(A, B)$ は単語 A から単語 B までの最小距離を求める関数であり，$w(q)$ では，係り受けインデックス q の係り元 ($l(q)$) と係り先 ($r(q)$) に対応する単語間の最小距離を求める際に用いている．これは，たとえ係り受け関係が文書中に含まれていなくても，係り受け関係を構成する単語同士が近くに出現していれば，それは検索クエリと関連する文書であろうという直感に基づいている．TSUBAKI では，係り先と係り元の距離が $D(= 30)$ 以下の場合，この直感に基づいたボーナスが与えられる．

5.3 検索の流れ

図 10 に検索の流れを示す．TSUBAKI は大きく，負荷分散サーバー（1 台），マスターサーバー（4 台），ファイルサーバー (1 台)，検索サーバー (43 台) から構成されており，検索サーバーの台数は同義表現を考慮した検索を行うかどうかにより異なる．具体的には，同義表現を考慮しない通常検索の場合 27 台，同義表現を考慮した検索の場合 43 台である．

TSUBAKI では 1 億ページを検索対象としているが，インデックスは 100 万ページ単位で管理されており，各インデックスデータは 27 台（同義表現の場合は 43 台）の検索サーバーに割り振られる．従って，各検索サーバーでは高々 400 万ページ分のインデックス（同義表現の場合は 300 万ページ分）から，検索クエリに適合するページを検索することになる．



1. 負荷分散サーバーにクエリ Q が与えられる
2. もっとも負荷の低いマスターサーバーにクエリ Q が送信される
3. クエリ Q を解析し，検索条件 C，単語インデックス Q_{word}，係り受けインデックス Q_{dpnd} を抽出する
4. C, Q_{word}, Q_{dpnd} を 27 台の検索サーバーに送信する
5. 各検索サーバーで C, Q_{word}, Q_{dpnd} に適合する文書を検索し，クエリとの適合度を表すスコアを計算する
6. 検索により得られたスコア付き文書の集合をマスターサーバーへ返信する
7. 各検索サーバーからの返信された検索結果をマージし，スコアに従いソートする
8. 上位 M 件について，タイトル，URL などをデータベースより獲得し，またファイルサーバーから標準フォーマットを読み込み，スニペッツを生成する
9. 検索結果をユーザへ提示する

図 10　検索の流れ（同義表現を考慮した場合は，検索サーバーの台数が 43 になる）

6　システム構成★

★どういったデータがローカルディスクのどこにどのくらいあるか pdf ファイルに追記する★ URL と ID の対応がどういったプログラムでどのように管理されているかドキュメントを書く

7　TSUBAKI API

本節では TSUBAKI が公開している API について述べる．

7.1　検索結果の取得方法

TSUBAKI では REST[3] 形式で，その API を公開している．API にアクセスするためのリクエスト URL は以下のとおりである．

http://tsubaki.ixnlp.nii.ac.jp/api.cgi

この URL の後ろに，表 7 に挙げたリクエストパラメータを付加し，HTTP にて問い合わせることで検索結果が得られる．以下にリクエスト URL の例を示す．

表 7　TSUBAKI API で利用可能なリクエストパラメータ一覧

パラメータ	値	説明
query	string	検索クエリ (utf8) を URL エンコードした文字列．検索結果を得る場合は必須．
start	integer	取得したい検索結果の先頭位置．
results	integer	取得したい検索結果の数．デフォルトは 10．
logical_operator	ANDOR	検索時の論理条件．デフォルトは AND．
only_hitcount	0/1	ヒット件数だけを得たい場合は 1，検索結果を得たい場合は 0．デフォルトは 0．
force_dpnd	0/1	クエリ中の係り受け関係を全て含む文書を得たい場合は 1，そうでない場合は 0．デフォルトは 0．
snippets	0/1	スニペッツが必要な場合は 1，スニペッツが不要な場合は 0．デフォルトは 0．
near	integer	クエリ中の単語と単語が n 語以内に出現するという条件のもと検索を実行する (近接検索)．クエリ中の単語の出現順序は考慮される．
id	string	個別の文書を取得する際の文書 ID．オリジナルのウェブ文書，または標準フォーマット形式の文書を得る際は必須．
format	htmlxml	オリジナルのウェブ文書，または標準フォーマット形式のウェブ文書のどちらを取得するかを指定．id を指定した際は必須．

■例 1: 「京都の観光名所」について検索した結果の上位 20 件を取得したい場合

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

ここで,

query: キーワードを URL エンコーディングした文字列 (この例の場合は「京都の観光名所」)
start: 検索結果の取得を開始したい検索結果中での順位 (オフセット)
results: 検索結果を取得したい start からの文書数

を意味する．

■例 2:「京都の観光名所」のヒット件数だけを知りたい場合

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&only_hitcount=1

この時，ヒット件数だけが plain/text 形式で返される．

図 11 に例 1 の場合について，perl を使って TSUBAKI API にアクセスするためのサンプルコードを示す．実際にこのコードを実行すると検索結果が XML 形式で出力される（図 12 参照）．表 8 に検索結果で使われているタグおよび属性を示す．

7.2　キャッシュされたウェブページおよび標準フォーマット変換済みデータの取得方法

キャッシュされているウェブページおよび，その文書を標準フォーマットに変換したデータについても以下のアドレスにアクセスすることで取得できる．

http://tsubaki.ixnlp.nii.ac.jp/api.cgi

```
#!/usr/bin/env perl

use LWP::Simple;
use URI::Escape;
use Encode;
use utf8;

my $query = encode('utf8', '京都の観光名所'); # 検索したいキーワード (utf8)
my $uri_escaped_query = uri_escape($query); # 検索キーワードを URL エンコーディングする
my $base_url = 'http://tsubaki.ixnlp.nii.ac.jp/api.cgi';
my $results = 20;
my $start = 1;
my $req_url = "$base_url?query=$uri_escaped_query&start=$start&results=$results&snippets=1";

# TSUBAKI API の結果を取得
my $response = get($req_url);
print $response;
```

図 11　API にアクセスするためのサンプルコード

表 8　検索結果に含まれるタグおよび属性

フィールド	説明
ResultSet	このタグで囲まれた部分が検索結果を表しており，次の属性を持つ． time: 検索を実行した日時 query: 検索キーワード totalResultsAvailable: 検索キーワードを含む文書数 totalResultsReturned: 返された文書数 firstResultPosition: 検索時に指定した，文書を取得する際の開始順位 (オフセット) logicalOperator: 検索時の論理条件 dpnd: 検索が係り受けを考慮して行われたかどうか 0: 係り受けを考慮せずに検索を実行 1: 係り受けを考慮して検索を実行 filterSimpages: 類似ページフィルタが適用されているかどうか 0: 類似ページフィルタが適用されていない 1: 類似ページフィルタが適用されている
Result	このタグで囲まれた部分が検索して得られた 1 文書の情報で，次の属性を持つ． ID: 文書の ID Score: 文書のスコア 文書 ID は，キャッシュされたウェブページおよび標準フォーマット変換済みデータを取得する際に必要になる．
Title	ページのタイトル
Url	ページの URL
Snippet	ページに含まれる検索クエリと関連する重要文
Cache	キャッシュされたウェブページに関する情報
Url	キャッシュされたウェブページの URL
Size	キャッシュされたウェブページの gzip 圧縮時のファイルサイズ

このアドレスに format および id オプションを付与して，API にアクセスする．例えば，文書 ID が 07423972 番の文書について，キャッシュされたウェブページを取得したい場合は，以下の URL を生成し，HTTP にてアクセスする．

http://tsubaki.ixnlp.nii.ac.jp/api.cgi?format=html&id=074239724

同様に，標準フォーマット変換済みデータが欲しい場合は，format の値を xml に変更することで取得可能である．

```
<ResultSet time="2007-02-21 13:43:58" query="京都の観光名所" totalResultsAvailable="19586"
 totalResultsReturned="10" firstResultPosition="0" logicalOperator="AND" dpnd="1" filterSimpages="0">
  <Result Id="24411919" Score="68.67424">
    <Title>関西探索</Title>
    <Url>http://www.kansaitansaku.com/</Url>
    <Snippet>
      このホームページは，京都を中心とした観光名所におとずれ，その感想を述べていくホームページです．この
ホームページは，京都を中心とした観光名所におとずれ，その感想を述べていくホームページです．京都，観光
名所，寺院，関西探索 ...
    </Snippet>
    <Cache>
      <Url>
        http://tsubaki.ixnlp.nii.ac.jp/index.cgi?URL= INDEX_NTCIR2/24/h2441/24411919.html&KEYS=%B5%FE%
C5%D4%A4%CE%B4%D1%B8%F7%CC%BE%BD%EA
      </Url>
      <Size>619</Size>
    </Cache>
  </Result>
  <Result Id="06832429" Score="64.16455">
    <Title>京都観光タクシー　京都の名園１</Title>
    <Url>http://kyoto-kankou.com/page02.htm</Url>
    <Snippet>
      京都旅行はタクシー利用がお勧め，観光タクシーで紅葉観光，観光タクシー京都名所，京都の穴場コース京
都観光タクシー，タクシー観光，京都，紅葉，京都観光，旅行，情報，桜，庭園，社寺，観光案内［京都庭園コー
ス］［京都の紅葉］［京都の名所動画］［京都の穴場］［宿プラザ］［スタッフ］［料金表］［お申し込み］所要
時間約５時間数ある京都観光名所の中でも，最も季節によって景観に変化の有る情緒豊かな場所です．所要時間約 ...
    </Snippet>
    <Cache>
      <Url>
        http://tsubaki.ixnlp.nii.ac.jp/index.cgi?URL= INDEX_NTCIR2/06/h0683/06832429.html&KEYS=%B5%FE%
C5%D4%A4%CE%B4%D1%B8%F7%CC%BE%BD%EA
      </Url>
      <Size>2152</Size>
    </Cache>
  </Result>

... 中略...

  <Result Id="26054757" Score="60.45654">
    <Title>Ｋｙｏｔｏ　Ｓｈｉｎｂｕｎ：京都の観光名所へのタクシー運賃</Title>
    <Url>http://www.kyoto-np.co.jp/kp/koto/taxi/unchin.html</Url>
    <Snippet>
      ＫｙｏｔｏＳｈｉｎｂｕｎ京都の観光名所へのタクシー運賃ＫｙｏｔｏＳｈｉｎｂｕｎ京都の観光名所への
タクシー運賃京都の観光名所へのタクシー運賃【ＪＲ京都駅からの概算】週刊京都
    </Snippet>
    <Cache>
      <Url>
        http://tsubaki.ixnlp.nii.ac.jp/index.cgi?URL= INDEX_NTCIR2/26/h2605/26054757.html&KEYS=%B5%FE%
C5%D4%A4%CE%B4%D1%B8%F7%CC%BE%BD%EA
      </Url>
      <Size>1209</Size>
    </Cache>
  </Result>
</ResultSet>
```

図 12　API より得られる検索結果

http://tsubaki.ixnlp.nii.ac.jp/api.cgi?format=xml&id=074239724

8　配布ツールの使い方★

8.1　ウェブページから標準フォーマットへの変換

■クローラの出力からのウェブページの切り出し

■ウェブページを標準フォーマットに変換する

- WWW2sf を cvs co する
- tool/scripts/make-standard-format.sh の workspace, toolpath 変数の値を変更
- tool/scripts/make-standard-format.sh を実行
- sh tool/scripts/make-standard-format.sh 10000 ページごとにまとめられた HTML ディレクトリへの絶対パス

■文書 ID の歯抜けができるのでつめる

- tool/scripts/rename-did.sh を使ってつめる
- tool/scripts/rename-did.sh の distdir, workdir の値を変換
- tool/scripts/rename-did.sh を実行
- sh tool/scripts/rename-did.sh 文書 ID 開始オフセット 標準フォーマットが納められたディレクトリ（複数のディレクトリを指定可）
- 端数はどこか一箇所に集めて再度文書 ID を振りなおす

■10,000 ファイルが納められていることを確認して tar czf する

8.2 インデックスの作成方法

■SearchEngine を cvs co する

- login
  - cvs -d :pserver:nobody@reed.kuee.kyoto-u.ac.jp:/share/service/cvs login
  - パスワード：kuro-language
- check out
  - cvs -d :pserver:nobody@reed.kuee.kyoto-u.ac.jp:/share/service/cvs co SearchEngine

■標準フォーマットからインデックスを抽出（並列実行可） scripts/make-index.sh を利用する．6、7 行目の変数の値を変更する．

```
1 #!/bin/sh
2
3 # 1 万ページ毎に tgz された標準フォーマットの塊からインデックスを抽出するスクリプト
4
5 # ★以下の値を変更すること
6 workspace=/tmp/mk\_tsubaki\_idx
7 scriptdir=\$HOME/cvs/SearchEngine/scripts
8
```

実行例：sh scripts/make-index.sh anywhere/x00000.tgz
x00000.tgz … 標準フォーマットを tgz したファイル（標準フォーマットは gzip で圧縮してあること）

■インデックスデータのリストをつくる　インデックスデータのリスト（どのマシンの、どこにあるか）を作成する ((gxp) + find コマンド)

```
% cat index.list
iccc011.crawl.kclab.jgn2.jp:/data/home/skeiji/mk_idx_syn_071229/i00835.idx.gz
iccc011.crawl.kclab.jgn2.jp:/data/home/skeiji/mk_idx_syn_071229/i00864.idx.gz
iccc011.crawl.kclab.jgn2.jp:/data/home/skeiji/mk_idx_syn_071229/i01699.idx.gz
iccc011.crawl.kclab.jgn2.jp:/data/home/skeiji/mk_idx_syn_071229/i02202.idx.gz
iccc015.crawl.kclab.jgn2.jp:/data/home/skeiji/mk_idx_syn_071229/i00164.idx.gz
iccc011.crawl.kclab.jgn2.jp:/data/home/skeiji/mk_idx_syn_071229/i03050.idx.gz
iccc011.crawl.kclab.jgn2.jp:/data/home/skeiji/mk_idx_syn_071229/i03634.idx.gz
iccc011.crawl.kclab.jgn2.jp:/data/home/skeiji/mk_idx_syn_071229/i03689.idx.gz
iccc040.crawl.kclab.jgn2.jp:/data/home/skeiji/mk_idx_syn_071229/i00108.idx.gz
iccc015.crawl.kclab.jgn2.jp:/data/home/skeiji/mk_idx_syn_071229/i00283.idx.gz
・・・
```

■インデックスをマージする (並列実行可)　scripts/merge-index.sh を利用する．6、7 行目の変数の値を変更する．

```
1  #!/bin/sh
2
3  # 1 万件ごとのインデックスデータを 100 万件単位にマージするスクリプト
4
5  # ★以下の値を変更すること
6  workspace=/tmp/mg\_tsubaki_idx
7  scriptdir=$HOME/cvs/SearchEngine/scripts
8
```

実行例：sh scripts/merge-index.sh 000 anywhere/index.list
index.list … さっき作ったインデックスデータのリスト

■インデックスをバイナリ化し、データベースを構築する　(並列実行可)　scripts/binarize-index.sh を利用する 6、7 行目の変数の値を変更する

```
1  #!/bin/sh
2
3  # インデックスデータをバイナリ化し、各種データベースを作成するスクリプト
4
5  # ★以下の変数の値を変えること
6  workspace=/tmp/bin_tsubaki
7  scriptdir=$HOME/cvs/SearchEngine/scripts
8
```

実行例：sh scripts/binarize-index.sh anywhere/000.idx.gz
000.idx.gz … 100 文書分がマージされたインデックスデータ

■作成されるデータ

インデックスデータ: idx000.word.dat, idx000.dpnd.dat
オフセット DB: offset000.word.cdb,offset000.dpnd.cdb, offset000.word.cdb.1 …
文書長 DB: 000.doc_length.bin
タイトル DB: 000.title.cdb
URLDB: 000.url.cdb

■URL、タイトルデータベースの構築　WWW2sf 内の tool/scripts/extract-url-title.perl を利用.
make_cdb.perl.
未整理.

■文書頻度データベースの再構築　文書頻度データベースの再構築が必要.
未整理.

■インデックスの追加　文書 DB、タイトル DB を以下の場所に置く

- iccc001:/data/home/skeiji/dbs/titledb
- iccc001:/data/home/skeiji/dbs/urldb

検索サーバーの適当な場所に以下のファイルを置く

- 文書長 DB: ZZZ.doc_length.bin,
- タイトル DB: ZZZ.title.cdb,
- URLDB: ZZZ.url.cdb,
- インデックスデータ: idxZZZ.word.dat, idxZZZ.dpnd.dat
- オフセット DB: offsetZZZ.word.cdb, offsetZZZ.dpnd.cdb, offsetZZZ.word.cdb.1 …

設定ファイルを書き換える.

■設定ファイル　/share10/WISDOM/conf/TSUBAKI.conf

インデックスの追加、検索ノードの故障などがあった場合は、上記のファイルを編集すれば、TSUBAKI の動作に反映される。

```
##################################################
# TSUBAKI の環境変数（★ utf8 で保存すること）
# $Id: configure,v 1.3 2008/02/10 08:17:45 skeiji Exp $
#
#
#
# 主な設定項目
# * 基本ツールのパス
```

```
# * キャッシュページ／標準フォーマット関係のパス
# * データベースのパス
# * 検索サーバー／スニペットサーバー関係（ホスト名、ポート番号等）
# * その他の設定
#
#
#
# 書式
# * KEY\tVALUE
# * KEY\tVALUE1,VALUE2,...（値が複数の時）
#   - サーバー関係の設定は特別な書式なので、各項目を参照のこと
# * 行頭が‘#’の行はコメント
#
##################################################


##################################################
# 基本ツールの設定（★要変更）
##################################################
TOOL_HOME /home/skeiji/local/bin
KNP_PATH /home/skeiji/local/bin
JUMAN_PATH /home/skeiji/local/bin
SYNDB_PATH /home/skeiji/tmp/SynGraph/syndb/i686
KNP_OPTIONS -postprocess,-tab
KNP_RCFILE /home/skeiji/.knprc
SYNGRAPH_PM_PATH /home/skeiji/tmp/SynGraph/perl


##################################################
# キャッシュページ／標準フォーマット関係の設定
##################################################

# キャッシュページ/標準フォーマット全体を置いてあるディレクトリへのパス（★要変更）
HTML_FILE_PATH /net2/nlpcf34/disk08/skeiji
ORDINARY_SF_PATH /net2/nlpcf34/disk08/skeiji
SYNGRAPH_SF_PATH /net2/nlpcf34/disk09/skeiji/sfs_w_syn


##################################################
# データベース関係の設定
##################################################

# TITLEDB/URLDB を納めたディレクトリへのパス（★要変更）
TITLE_DB_PATH /work/skeiji/titledb
```

```
URL_DB_PATH /work/skeiji/urldb

# 単語・係り受け・同義表現・同義表現係り受けの DFDB を納めたディレクトリへのパス（★要変更）
ORDINARY_DFDB_PATH /var/www/cgi-bin/dbs/dfdbs
SYNGRAPH_DFDB_PATH /data/dfdb_syngraph_8600




##################################################
# 検索サーバー/スニペットサーバー関係の設定
##################################################

# 検索用サーバー（★要変更）
# SEARCH_SERVERS ホスト名 ポート番号 1,...

SEARCH_SERVERS  nlpc06 22001,22002,22003,20006
SEARCH_SERVERS  nlpc07 22001,22002,22003,20006
...
SEARCH_SERVERS  nlpc31 22001,22002,22003
SEARCH_SERVERS  nlpc32 22001,22002,22003

# SYNGRAPH 検索用サーバー（★要変更）
# SEARCH_SERVERS_FOR_SYNGRAPH ホスト名 ポート番号 1,...

SEARCH_SERVERS_FOR_SYNGRAPH nlpc06 50001,50002
SEARCH_SERVERS_FOR_SYNGRAPH nlpc07 50001,50002
...
SEARCH_SERVERS_FOR_SYNGRAPH nlpc47 50001,50002
SEARCH_SERVERS_FOR_SYNGRAPH nlpc48 50001,50002

# スニペットサーバーの設定（★要変更）
# STANDARD_FORMAT_LOCATION ホスト名 ポート番号 ディスクにおいてある標準フォーマットの ID1,...

STANDARD_FORMAT_LOCATION nlpc33 35000 000,016,032,048,064,080,096
STANDARD_FORMAT_LOCATION nlpc34 35000 001,017,033,049,065,081,097
...
STANDARD_FORMAT_LOCATION nlpc47 35000 014,030,046,062,078,094
STANDARD_FORMAT_LOCATION nlpc48 35000 015,031,047,063,079,095




##################################################
# その他の設定
##################################################

# 強調表示に用いる色
```

```
HIGHLIGHT_COLOR ffff66,a0ffff,99ff99,ff9999,ff66ff,880000,00aa00,886800,004699,990099;

# ログファイル（★要変更）
LOG_FILE_PATH /se_tmp/input.log

# マシンの故障等でサービスを停止する場合は 1
SERVICE_STOP_FLAG 0

# 検索画面に表示するメッセージ
MESSAGE 0

# スニペットの長さ（デフォルトは 100 単語）
MAX_NUM_OF_WORDS_IN_SNIPPET 100

# タイトルの長さ（デフォルトは 60 バイト）
MAX_LENGTH_OF_TITLE 60

# ID からキャッシュページへのパスを求める際のテンプレート（★要変更）
CACHED_HTML_PATH_TEMPLATE /net2/nlpcf34/disk08/skeiji/h%03d/h%05d/%09d.html.gz

# ID から標準フォーマットへのパスを求める際のテンプレート（★要変更）
SF_PATH_TEMPLATE /net2/nlpcf34/disk08/skeiji/x%03d/x%05d/%09d.xml.gz

# index.cgi のアドレス（★要変更）
INDEX_CGI http://tsubaki.ixnlp.nii.ac.jp/index.cgi

# キャッシュページへのリンクを生成する際に用いるテンプレート
CACHED_PAGE_ACCESS_TEMPLATE cache=%09d
```

8.3 検索プログラムの実行

■検索 CGI の設置

■TSUBAKI サーバーの起動手順

1. gxp を起動
2. 検索サーバーで使うノードを獲得

   edges -> iccc01[1,2,3,4,5,6,7,8,9]
   edges -> iccc02?
   edges -> iccc03[1,2,3,4,5,6]
   explore # iccc011 - iccc036 のノードを獲得

3. プログラムがあるディレクトリに移動

   cd /share09/home/skeiji/cvs/SearchEngine/scripts

4. プログラムで使用できるメモリの上限を解除

e ulimit -Ss unlimited

5. 検索サーバプログラムの起動

   e perl -I ../cgi -I /tmp/SynGraph/perl tsubaki_server.pl -idxdir /data/home/skeiji/dat ★ -dlengthdbdir /dat/home/skeiji/dat ★ -port 5000 ★

9 おわりに

検索エンジン基盤 TSUBAKI では，独自に開発された検索エンジンコンポーネント (例えば，検索結果のランキングモジュールやリンク解析技術など) との融合を歓迎している．独自のコンポーネントを TSUBAKI に組み込みたい方は別途ご相談頂きたい．計算サーバーの提供を検討する．

参考文献

[1] Tomohide Shibata Michitaka Odani Jun Harashima Takashi Oonishi Sadao Kurohashi. Syngraph: A flexible matching method based on synonymous expression extraction from an ordinary dictionary and a web corpus. In *Proceedings of Third International Joint Conference on Natural Language Processing (IJCNLP2008)*, 2008.

[2] 新里圭司 橋本力 河原大輔 黒橋禎夫. 自然言語処理基盤としてのウェブ文書標準フォーマットの提案. In **言語処理学会 第** *13* **回年次大会** *(NLP2007)*, pages 602–605, 2007.

[3] Roy Thomas Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine, 2000.

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

[5] Sadao Kurohashi and Makoto Nagao. A syntactic analysis method of long japanese sentences based on the detection of conjunctive structures. *Computational Linguistics*, (4):507–534, 1994.

[6] Stephen E. Robertson, Steve Walker, Micheline Hancock-Beaulieu, Aarron Gull, and Marianna Lau. Okapi at TREC. In *Text REtrieval Conference*, pages 21–30, 1992.