

オープンソースを利用した 共出現単語情報データベース構築の試み

中 尾 泰 士 *

概 要

文書を主たる対象とする検索システムには、検索者からいくつかの検索語を受け取り、システム側であらかじめ作成しておいたインデックスを利用して、その語を含む「情報」を検索結果として返すものが多い。このようなシステムでは、網羅性は高いが、検索者が求めていない不要な情報までも大量に拾い出す傾向にあることが指摘されている。その欠点を補うためには、検索者に適切な検索語（索引語）を選択させたり、大量に返される情報から絞り込みを行えるような検索支援方法をシステム側で用意しておかねばならない。そのような検索支援方法の一つとして、検索者が入力した検索語と共によく使われている語の一覧を表示して、絞り込みを支援する方法が考えられている。本論文は、著名なオープンソースソフトウェア“Namazu”を用いて、そのような検索支援システムに必要な単語間の共出現頻度を計算し、データベース化する試みについて述べたものである。

1. はじめに

1.1 インターネット上の検索システム

情報の大部分が、書籍を中心としたアナログの形で流通、蓄積されていた時代には、その情報にアクセスするために、検索者の一定の知識や技能、および、努力を必要としてきた。そのような時代においては、情報検索は一部の人達しか行わない特殊なことだったと言えよう。

しかし、現在では、コンピュータやネットワーク、および、情報のデジタル化の普及によって、情報検索は比較的手軽に行えるようになってきている。ただ、それは必ずしも、一般の検索者にとって親切なものばかりとは言えない。情報検索が身近になったおかげで、いかに検索者の知識や技能不足を補い、検索を支援するかという研究が、検索システムに関する一つの研究分野として重要な位置を占めるようになっている。情報検索分野における技術の変化は激し

* 奈良産業大学経営学部

中 尾 泰 士

いものがあるが、その技術動向については、特許庁のサイトに掲載されている「標準技術集：サーチエンジン」が参考になるだろう^[1]。

現在、情報検索が最も必要とされているのは、インターネット上の検索であろう。日々増大し続けるサイトやページから、必要な情報を効率良く取得するには、検索サイトを使用せずにほぼ不可能とも言える。インターネット上の代表的な検索サイトとしては、ディレクトリ型のYahoo! Japan^[2] (Yahoo!^[3] の日本版)、ロボット型のGoogle^[4] などが有名である。

ディレクトリ型とは、インターネット上のサイトをカテゴリ別に分類して、検索者に提供するものなどを指す。システム提供側が、あらかじめ検索対象となるサイトやページをふるいにかけ、人手をかけて適切なカテゴリに分類しておくわけだ。途中に、人間の価値判断が介在するため、検索結果の信頼性が高く、検索者の求める情報が比較的手に入りやすい特徴がある。この特徴から、ディレクトリ型の代表的な検索サイトであるYahoo! Japanの日本での利用率は高い。例えば、利用検索エンジンの2001年3月調査^[5]では、回答者1,684人のうち61.6%がYahoo! Japanを利用しているとの結果が出ており、2位の goo^[6] (31.9%)、3位の infoseek Japan^[3] (18.0%) を大きく引き離している。この調査では、本家の Yahoo!^[3] も14.2%で4位につけており、Yahoo! のブランド力は大きなものがあるようである^{注1}。

一方、ロボット型検索サイトにおいては、人手はかけず、プログラムによる自動処理を基本にしている。そのため、膨大な量の情報を処理するのに適している。インターネット上を仮想的に動き回る「ロボット」と呼ばれるプログラムが、ページ情報を収集し、集めた情報をあらかじめ索引語ごとにインデックス化しておく。そして、検索者から索引語とともに検索要求が出された時に、用意しておいたインデックスから該当ページ情報を返すというのがロボット型検索システムである。これは基本的に全文検索システムになっている。

ロボット型検索システムでは、与えられた索引語を含むページを網羅的にピックアップしてしまうため、返される検索結果が大量になりやすいという欠点がある。そのため、商用の検索サイトではアルゴリズムを工夫して、重要なページ順にソートした形で検索結果を返すなどの努力をしている。

たとえば、Google^[4]では、該当ページの被リンク情報（他のページからリンクされている情報）をもとにページの重要度を判定していると言われる。簡単に説明すれば、他のページからのリンクを当該ページに対する「支持投票」と見なししてランクづけするのである。かつ、この評価システムにおける「重要な」ページからのリンクには、それにふさわしい重みを与えていているという。このページランクシステムについての詳細は、Google自身による「Googleの人気の秘密^[8] や、基本的な文献^{[9] [10]}などを参照されたい。

Googleは、試験段階の頃からNetscape社の検索システムと提携していたのをはじめ、2000

^{注1} この調査では、後述する Google は初登場で、4.9%の11位となっている。

オープンソースを利用した共出現単語情報データベース構築の試み

年には Yahoo! とも提携するなど、現在では、多くの検索システムのバックエンド（各社のシステムで検索出来ない場合に補完するシステム）として動いている。2002年4月の米調査会社 WebSideStory^[11]によれば、検索エンジン市場での世界シェア上位は、Yahoo! 36%、Google 32%、MSN^[12] 13% ということである^{注2}。Googleについて言えば、2000年6月の約1%から2年弱で32%にシェアが急上昇しているとの調査結果であり、Googleが急速に受け入れられている様子がうかがえる。

Google とディレクトリ型のYahoo! が提携していることでも分かるように、ディレクトリ型とロボット型は、検索システムとして相互に補完し合う関係にあると言え、現在では検索サイトの構成に関する限り、両者の境目は必ずしも明確ではない。その他、複数の検索サイトに検索要求を出し、結果を結合して返すメタ検索サイト（例えば、metacrawler^[13] やMetcha Search^[14]など）や、「ガイド」と呼ばれる案内人に、有用なサイトを紹介してもらう、All About Japan^[15]のような新しいタイプの検索サイトも出現するなど、検索システムにおけるサービスの拡充は多岐にわたっている。

1.2 AltaVista の Refine 機能

検索結果を絞り込んだり、再検索を支援する面白い試みとして、英語のみのサービスではあるが、AltaVista^[16]のRefine機能“Prisma”がある。これは、検索者が与えた語で検索を行ったときに、その検索語に関連する単語の一覧が表示されるというものである。

図1は、単語“iraq”で検索を行った例である（2002年9月25日検索）。“iraq”を含むページは、300万ページもあったのだが、検索結果とともに、システムが“iraq”に関連すると判断した単語：

Ancient, Baghdad, Campaign, Democracy, Desert Storm, Government, Iraq Action Coalition, Iraqi, Mass Destruction, Resolution, Sanctions, Weapons

が表示されている。ユーザはこの関連語一覧から、自分が調べたい項目を選択クリックすれば、絞り込み検索が出来るという仕組みである^{注3}。

^{注2} WebSideStory社のWeb利用動向分析サービス“StatMarket”によるもので、世界の12万5,000サイト以上で、1日に5,000万人以上のユーザから収集したデータを分析しているという。前述のインターネット協会による調査とは、対象や時期、方法が異なるので単純には比較できない。

^{注3} AltaVista の説明（Help）によれば、この Refine 機能は連続2回まで行えるとのことである。また、検索結果が20ページ以下の場合には、Refine 機能は使えないようだ。

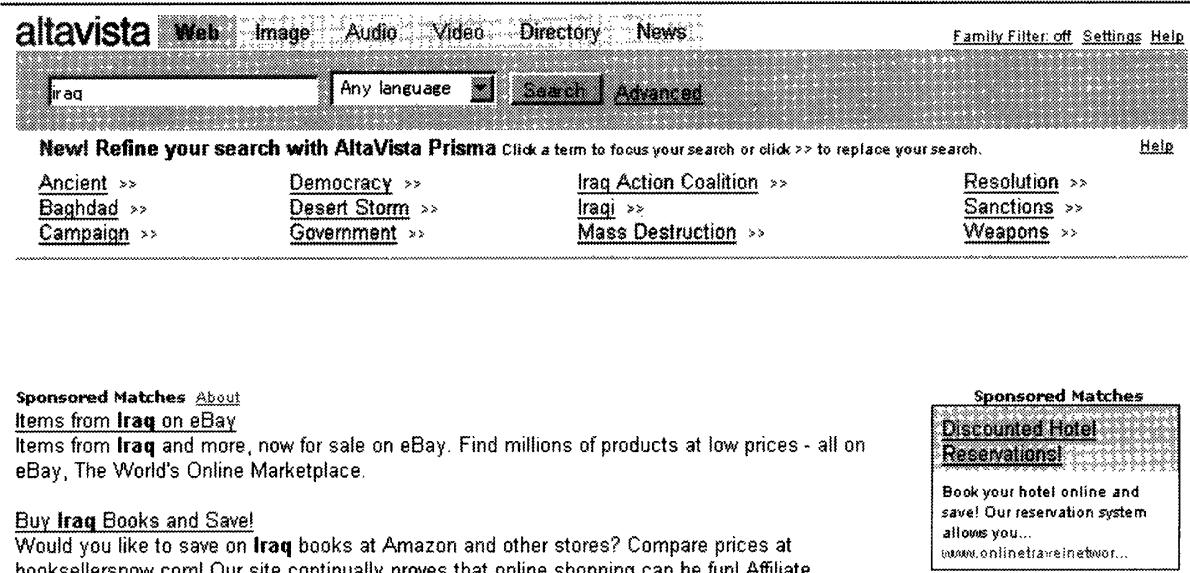


図1：AltaVistaで、単語“iraq”で検索を行った例。多くの検索結果とともに、“iraq”に関連した単語の一覧が表示される。

インターネット上の情報検索サービスについて、1998年に比較検討を行っている村上^[17]によれば、AltaVistaのRefine機能ではグラフ形式での関連語表示も出来たとのことである。しかし、現在（2002年9月）では、グラフ形式の表示は行えないようだ。

どのようにして関連語を選択しているのか、AltaVistaのアルゴリズムは公開されていないが、大量のデータを統計的に処理し、単語間の共起関係を基礎にしているのではないかと推測することは出来る。関連語を使った絞り込み検索で返されるページには、最初に用いた検索語と、Refine機能で選択した関連語が、同時に含まれるはずだからである。

本論では、このような単語と単語の間の共起関係について調べるツールをオープンソースを利用して作成したことについて述べる。なお、以下では、単語間の共起関係について、同じ文書に共に出現するということで「共出現」と呼ぶことにする。

2. 基本コンセプト

2.1 単語の共出現

N 個の文書からなる文書集合 A を考えよう。その中の文書に識別番号 i をつけ、それぞれ T_i とする：

$$T_i \in A \quad (i = 1, 2, \dots, N) . \quad (1)$$

さらに、個々の文書に含まれる単語を一定の条件によって抽出し、索引語として登録する。そうして出来た、文書集合 A に含まれる索引語の集合を K_A 、その要素を w_j とする：

$$w_j \in K_A \quad (j = 1, 2, \dots, M_A) . \quad (2)$$

ここで、索引語集合の要素の数を M_A とした。この要素の個数も含め、索引語集合 K_A の要素は、文書集合 A が異なれば、当然異なる集合となる。

さて、文書集合 A の中の文書 T_i には、索引語集合 K_A に含まれる、いくつかの索引語が含まれているはずである。つまり、文書 T_i に含まれる索引語の集合 K_{T_i} を作れば、 K_{T_i} は、索引語集合 K_A の部分集合になっている。そして、文書 T_i に索引語 w_j が含まれるかどうかをすべての文書、すべての索引語について調べれば、

$$\begin{cases} a_{ij} = k_c, & \text{if } w_j \in K_{T_i} \subseteq K_A \\ a_{ij} = 0, & \text{otherwise,} \end{cases} \quad (3)$$

という $N \times M_A$ 行列 $D_A = \{a_{ij}\}$ を作ることが出来る。ここで、 k_c は、索引語 w_j が文書 T_i に含まれる回数をカウントした正の整数が入るものとする。

2.2 索引語による文書検索

(3)式で考えた行列 D_A をもとにして、検索語として与えられた索引語を含む文書を検索することを考えよう。与えられた検索語が索引語 w_j である場合^{注4}、行列 D_A の j 列を見て、0以外の数値（この場合は、正の整数）を持つ行番号 i を抽出していくべき。それから、文書 T_i を識別することが出来る。

では、複数の索引語によるAND検索の場合はどうなるだろうか。たとえば、索引語 w_j と w_k ($j \neq k$) が与えられ、その双方を含む文書を検索したいとしよう。この場合、行列 D_A の i 行目の j 列と k 列の要素を掛け合わせ、

$$a_{ij} a_{ik} \neq 0, \quad (4)$$

となる行番号 i に相当する文書 T_i が、検索に適合する文書ということになる^{注5}。

2.3 単語の共出現指数

さらに、(4)式からは別の情報が引き出せる。すなわち、すべての文書番号 i について a_{ij} a_{ik} の値の和をとれば、索引語 w_j と w_k の間の結びつきの強さを示す指標として利用することが出来る。この指標を索引語同士の「共出現指標」と呼び、索引語 w_j と w_k の間のそれを c_{jk}

^{注4} もし検索語が索引語集合 K_A に含まれないものであった場合、行列 D_A を調べるまでもなく、この文書集合 A には、その索引語を含む文書はない。

^{注5} 物理学などにおける記法では、同じ添え字がある場合、「縮約」の意味になるが、(4)式では縮約はとらない。

と表現することにしよう：

$$c_{jk} \equiv \sum_{i=1}^N a_{ij} a_{ik}. \quad (5)$$

索引語集合 K_A に含まれる、すべての索引語のペアについて、この指標を計算すれば、 $M_A \times M_A$ の正方行列 $C = \{c_{jk}\}$ が構成される。当然ながら、この行列は対称行列で、かつ、そのすべての対角成分は0ではない：

$$c_{jk} = c_{kj}, c_{jj} \neq 0, (j, k = 1, 2 \cdots, M_A) \quad (6)$$

3. 実 装

3.1 全文検索エンジンNamazuのしくみ

第2節の基本コンセプトをもとに、索引語の共出現指標を算出し、データベース化するツールを作成した。その際、オープンソースの全文検索エンジン「Namazu」^[18]を利用している。ここで、Namazuについて、その動作原理を筆者が理解している範囲で簡単に述べておこう。

すでに述べたように、全文検索エンジンは、与えられた文書集合から、索引語を抽出し、どの文書にどの索引語が含まれるかをインデックス化する。その際、英語などの言語では、語と語との間に空白が空けられるため、コンピュータは簡単に語を識別して、索引語データベースを作成することが出来る。

しかし、日本語のように語と語をつなげて書く言語においては、この索引語の抽出は容易ではない。文章の中での語の認定（いわゆる「分かち書き」）を自動的に行う必要があり、情報言語処理の分野では「形態素解析」（morphological analysis）と呼ばれる大きな研究分野となっている（たとえば、文献^[19]などを参照）。

Namazuは、この分かち書きをKAKASIと呼ばれるプログラムを用いて行っている^{注6}。KAKASIは内部の「辞書」を利用し、その「辞書」に登録されている語に基づいて分かち書きを行う。この方法による分かち書きの精度は、内部の「辞書」の性能に大きく依存する。すなわち、「辞書」に登録されていない単語は、語として認識することが出来ない。これは特に、複合語などを正しく分かち書き出来ない等の欠点になってあらわれる。

さて、Namazuは文書を分かち書きにし、その文書中に含まれる索引語を抽出して、インデックス化していく。そして、作成されたインデックスは、Namazuの本来の目的である全文検索の際に使用されるのである。Namazuについての詳しい仕組みや、使い方などは、Namazuプロジェクトの一員である馬場氏による包括的な解説があるので、そちらを参照されたい^[20]。なお、Namazuには、KAKASIの他に、奈良先端大学院大学の松本研究室で開発されている

^{注6} KAKASIは、もともとは漢字かな混じり文をひらがな文やローマ字文に変換する目的で作成されたプログラムと辞書の総称で、GPLに基づくフリーソフトウェアである。

オープンソースを利用した共出現単語情報データベース構築の試み

「茶筅」という形態素解析プログラム^[21]を使用するオプションも用意されていることを付言しておく。

3.2 索引語の共出現指數を計算するツール

今回、作成したツールは、Namazuによって作成されるインデックスファイル群を利用して、索引語の共出現指數を計算するものである。具体的には、Namazuによって作成される多くのインデックスファイル群のうち、“NMZ.w”と“NMZ.i”を使用する。Namazuはバージョン2からインデックスの構造が大きく変わったが、ツール作成に当たって参考にしたNamazuのバージョンは2.0.5である。

インデックスファイル“NMZ.w”は、文書群から抽出された索引語の一覧を格納している。このファイルはテキストファイルなので、エディタ等で開いて中身を見ることが出来る。一方、“NMZ.i”には、索引語がどの文書に含まれているかなどの情報が含まれている。しかし、このファイルはバイナリファイルのため、エディタで開いても中身を見ることが出来ない。

ところが幸いなことに、オープンソースであるNamazuは、プログラムのソースを参照できるため、どのような形式でバイナリ化されているかを調べることが出来る。具体的には、インデックス作業を行うPerlスクリプトプログラム“mknmz”を解析すればよい。

最初に行う作業は、“mknmz”的ソースを参考にして、インデックスファイル“NMZ.w”と“NMZ.i”から、索引語とそれが含まれる文書番号（文書の識別番号）、その索引語がその文書に含まれる頻度などの情報をテキストファイルとして復元することである。これは一般には、

$$w_i, N_1, f_1, dN_2, f_2, dN_3, f_3, \dots,$$

のような書式の「行」を索引語 w_i の数だけ持つファイルとなる。ここで、 N_1 は、索引語が含まれる最も若い文書番号である。 dN_2 は、次にその索引語が含まれる文書番号までの増分である。すなわち、2番目にその索引語を含む文書の識別番号は、 $N_2 = N_1 + dN_2$ となる。以下同様に、3番目にその索引語を含む文書の識別番号は、 $N_3 = N_2 + dN_3$ のようになる。一方、 f_i ($i = 1, 2, 3, \dots$)は、その索引語が番号 N_i の文書に含まれる頻度である^{注7}。

こうして復元されたインデックス情報は、実は、(3)式で考えた行列 D_A の別の表現に他ならない。この行列をもとに、(5)式で考えた、索引語 w_i と w_j との間の共出現指數 c_{ij} を計算することが出来る。そして、

$$w_i, w_j, c_{ij},$$

の書式で、データベースを作成する。このデータベースの「行」は、索引語の数が M_A だとす

^{注7} 正確には、Namazuの仕様で、HTMLのタグ情報などに基づく一定の重みをつけられた整数である。

中 尾 泰 士

ると、 $M_A C_2 = M_A(M_A - 1)/2$ になる。これは、索引語の数が2倍になれば、データベースを構築するための計算量がおよそ4倍になることを示している。

しかし、実際には、Namazu を使って抽出された索引語には、およそ索引語とは呼ばれないものも含まれているため^{注8}、共出現指數を計算する前に、Namazu が作ったインデックスに適当なフィルタをかけて、索引語の数を減らす方が効率的である。どのようなフィルタリングが適当かは、慎重な検討が必要だと思われるが、ここでは試験的に次のようなフィルタリングを行った：

1. 全角5文字以上の単語を除外する。
2. ひらがなののみの単語を除外する。
3. ひらがなで終る単語を除外する。
4. 英数字のみの単語を除外する。
5. その他、あらかじめ不要リストに登録された単語は除外する。

最後のフィルタリング項目にある不要リストとは、文書群中のあまりにも多くの文書に含まれていて、個々の文書の特徴としては適切ではない単語の一覧リストである。

さらに、ほんの数回、共出現する索引語のペアをデータベースに登録するのを避けるため、一定の閾値を超える頻度で共出現するペアのみをデータベースに登録することにした。

3.3 ツールの適用結果サンプル

以上のような考え方で作成したツールを奈良産業大学図書館（以下、「奈良産大図」）の蔵書データに適用し、共出現指數のデータベースを構築してみた。図書館蔵書データを使用したのは、以下のような理由による：

1. 「奈良産大図」のWebOPAC (Web Online Public Access Catalog) は、Namazu をもとにして、筆者ら^[22]によって構築されていて、インデックスファイルが手近にあったため。
2. 図書館の蔵書データは、およそ15万件という大きな量であることと、かつ、形式が整った同質の文書の集合体であるため。
3. 将来的には、蔵書検索システムに応用しようと考えているため。

「奈良産大図」の個々の蔵書データファイルの構造については、文献^[23]を参照して欲しい。ただ、蔵書データという性格から、ほとんどすべてのファイルには、

“著”，“共著”，“印刷”，“改訂”，“出版”，“編”，“編著”，“訂”

^{注8} 例えば、「する」「ある」などの動詞や「を」「は」などの助詞。

オープンソースを利用した共出現単語情報データベース構築の試み

などの単語が含まれているため、これらを不要単語リストとして登録し、共出現指數の計算から除外している。

表1に、このデータベースを用いて得られた、共出現指數の高かった単語のペアをサンプルとしていくつかあげておく。

検索語	共出現指數が高い語
「法廷」	→ 「弁護」
「吉野」	→ 「山桜」
「地方」	→ 「人口」
「公務員」	→ 「地方」「東京」
「保護」	→ 「法理」
「漢字」	→ 「北京」
「権利」	→ 「擁護」

表1：共出現指數の高い単語のペア例。左の単語を固定したときに、共出現指數が目についた単語をあげた。

では、本論で述べたような共出現指數にはどのような利用価値があるのだろうか。第4節では、その利用法をいくつかあげてみることにする。

4. 共出現データベースの応用と今後

4.1 複合語の検出と辞書への自動追加

ある単語 w_i と単語 w_j の間の共出現指數 c_{ij} が大きな値をとった場合、その2つの単語の間には、強い「関係」があると判断できるだろう。例えば、「電子」という単語と「商取引」という単語が大きな共出現指數を持つ場合、一つの可能性として、「電子商取引」という複合語を考えられる。

既述したように、日本語全文検索システムでは、文章を単語に分解するために「辞書」を使用することが多い。そして、文章を分解する能力は、そのシステムが持つ「辞書」の性能に大きく依存するのである。日本語の特徴の一つとして、単語と単語を結びつけて新たな複合語を作り出されるという点があるが、そのような新たに作り出されたキーワードに対しては、静的な「辞書」ではうまく対処できないことになる。

そこで、共出現指數を利用して、複合語の可能性のある単語の組み合わせを自動的に検出し、「辞書」に登録すれば、「辞書」の動的な変化を促すシステムを作成することが可能になるだろう^{注9}。

^{注9} ここでいう「辞書」とは、われわれが普段使用しているような「正しい」ことが書かれている辞書である必要はない。先の例で言えば、「電子」と「商取引」の共出現指數が高い場合は、「電子商取引」と「商取引電子」の2つを辞書に登録すればよい。もちろん、後者は「正しくない」単語であるが、この正しくない言葉が実際の文書に出る可能性はほとんどないため、文章分割の処理には何ら影響を与えないであろうからである。

4.2 シソーラス

仮に、索引語 w_i と w_j との間に大きな共出現指数 c_{ij} があり、また、索引語 w_i と w_k との間にも同様に大きな共出現指数 c_{ik} があるとしよう。かつ、このとき、索引語 w_j と w_k との間の共出現指数 c_{jk} が小さかったとしよう（図2参照）。これは、索引語 w_i と w_j 、 w_i と w_k はよく一緒に現れるが、 w_j と w_k はめったに一緒に現れない状況をあらわしている。この場合、一つの可能性として、索引語 w_j と w_k は類義語であることが考えられる。

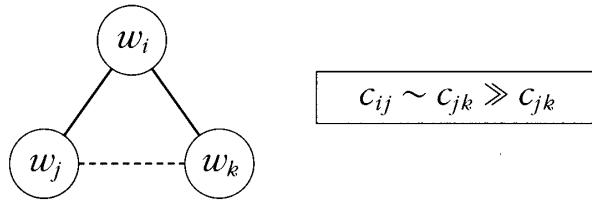


図2：同値関係にある索引語。語 w_i と w_j 、語 w_i と w_k との間の共出現指数は大きいが、語 w_j と w_k の間の指数は小さい場合、語 w_j と w_k はよく似た意味の語である可能性がある。

例えば、「関係」と「親類」と「親戚」の3つの語を考えよう。「関係」と「親類」、「関係」と「親戚」の各ペアは同一文の中に出現する可能性が高いが、類義語である「親類」と「親戚」を同時に使用する可能性は低いと予想される。このことから、図2のような状況にある3語を検出することで、シソーラスを半自動的に作成できる可能性がある。

4.3 検索支援システムへの応用

先に紹介した AltaVista の Refine 機能は、絞り込みを行う支援システムとして有用である。そもそも、全文検索システムでは、索引語として登録されていない検索語では検索できないのであるから^{注10}、検索語をユーザに追加させる形の絞り込み検索よりも、システム側から、

「あなたの検索した語を含む文書には、このような語が同時に含まれています。」

という、共出現情報が提供されれば、ユーザの負担は軽減されるだろう。将来的には、「奈良産大図」の蔵書検索システム中に、このような検索支援システムを組み込めないかと考えている。

5. 終わりに

本研究の源は、筆者らが奈良産業大学図書館のWebOPACシステムを構築した際に^[22]、その検索エンジン部分としてオープンソースの全文検索エンジン Namazu を採用したことにある。その時の経験から、一つには、全文検索という性格から来る検索ごみをうまく取り除いて、検

^{注10}もちろん、システム側にシソーラスが用意されていれば、その限りではない。

オープンソースを利用した共出現単語情報データベース構築の試み

索効率を上げるために、何か良い支援システムはないかと考えはじめたこと、そして、もう一つには、「オープンな」Namazu のソースをのぞいて、その動きに興味を持ったことが本論の始まりである。

第4.3節で考察したように、検索精度を高めるためには、検索途中に表示される関連語を使って、検索者を適切な方向に導く手法が効果的である。たとえば、「保守」という検索語に対して、「政治」や「電気」などの、「保守」とともによく使われる単語を提示してやるのである。そうすれば、「保守政治」に関心のある検索者は、「電気工事」の情報に煩わされることがなくなり、逆もまた然りである。

そのような考えから、筆者は索引語の共出現関係に着目したのだが、その共出現関係を調べるツールの一部に Namazu を利用することによって、最も難しい、日本語文章を単語へ分割する部分を省くことが出来、ツール開発の時間を大幅に短縮することが出来た。これはオープンソースの大きな利点の一つである。

情報学において、情報検索の分野はいま最も注目されている領域であり、本論で述べたようなトピックスについては、情報工学や自然言語処理論などの様々な方面で盛んに研究が行われている（たとえば、文献^[19]^[24]などを参照）。筆者は、そのすべてをフォローできていないし、第4節で述べた応用についても、まだ構想の段階にとどまっている。

しかし、誰にでも入手可能なオープンソースでツールを作成したことに、本研究の意義があると考えている。すなわち、ある特定のシステムや環境でのみ使用可能なツールではなく、興味のある人は誰でも使って試してみることが出来るツールということを目指している。

また、Namazu という一般の多くの人が使用しているアプリケーションを利用していることため、Namazu の利用者からインデックスファイルを提供してもらえば、開かれた参加型のプロジェクトとして発展する可能性もある。コンピュータの世界でよく使われる言葉に、“Garbage in, garbage out.” というものがあるが、多くの人から集めた情報には、そこに何らかの有用な情報が含まれている可能性が高いのではないだろうか。多数の人の参加によって、単語の共出現という観点から、言葉のネットワークが形成できたら面白いと考えている。

本研究にあたっては、奈良文化女子短期大学図書館の村上幸二氏に、文献資料の教示から、データベース登録作業まで、多大なる支援をいただいた。表1にあげた共出現単語のサンプルも村上氏の協力により、抽出できたものである。ここに感謝したい。

参考文献およびURL

- [1] 特許庁「標準技術集：サーチエンジン」,
http://www.jpo.go.jp/techno/hyoujun_gijutsu/search_engine/
- [2] Yahoo! Japan, <http://www.yahoo.co.jp/>
- [3] Yahoo!, <http://www.yahoo.com/>
- [4] Google, <http://www.google.com/>
- [5] インターネット協会監修「インターネット白書2001」、第3章 パソコン利用者「検索エンジン/ポータルサイト」、p.70 インプレス、2001年
- [6] goo, <http://www.goo.ne.jp/>
- [7] infoseek Japan, <http://www.infoseek.co.jp/>
- [8] 「Googleの人気の秘密」, http://www.google.co.jp/intl/ja/why_use.html
- [9] Lawrence Page, Sergey Brin, Rajeev Motwani, Terry Winograd, “The PageRank Citation Ranking: Bringing Order to the Web” , 1998
<http://www-db.stanford.edu/textasciitilde{}backrub/pageranksub.ps>
- [10] Taher H. Haveliwala, “Efficient Computation of PageRank” , Stanford Technical Report, 1999,<http://www-db.stanford.edu/textasciitilde{}taherh/papers/efficient-pr.ps>
- [11] “Google Challenges Yahoo as the No.1 Search Site in the World, According to WebSideStory's StatMarket” (2002年9月23日アクセス),
http://www.websidestory.com/cgi-bin/wss.cgi?corporate&news&press_1_177
- [12] MSN, <http://www.msn.com/>
- [13] metacrawler, <http://www.metacrawler.com/>
- [14] Metcha Search, <http://bach.scitec.kobe-u.ac.jp/metcha/>
- [15] All About Japan, <http://allabout.co.jp/>
- [16] AltaVista, <http://www.altavista.com/>
- [17] 村上晴美、「WWW上の知的情報検索－現状の問題点と解決へのアプローチー」、日本図書館情報学会研究委員会編「情報検索の理論と実際」所収、p.80－106、日外アソシエーツ、1999年
- [18] 全文検索エンジン Namazu, <http://www.namazu.org/>
- [19] 徳永健伸, 「情報検索と言語処理」, 東京大学出版会、1999年
- [20] 馬場肇, 「Namazu システムの構築と活用」、ソフトバンク、2001年
- [21] 形態素解析システム 茶筌 (ChaSen), <http://chasen.aist-nara.ac.jp/>、および、松本裕治、北内啓、山下達雄、平野善隆、松田寛、高岡一馬、浅原正幸、「形態素解析システム『茶筌』version 2.2.1 使用説明書」、奈良先端科学技術大学院大学 松本研究室, 2000年
- [22] 中尾泰士、村上幸二、「オープンソースによる簡易WebOPACの開発－全文検索ソフト Namazu の応用」、情報管理、vol.44、No.6、p.412－420、2001年
- [23] 中尾泰士、「オープンソースによるシステム構築の意義と可能性」、産業と経済、vol.16、No.5、p.35－50、2001年
- [24] 長尾真、宇津呂武仁、島津明、勾坂芳典、井口征士、片寄晴弘、「文字と音の情報処理」、岩波書店、2000年