5

Namazu：全文検索で文書の山に立ち向かう

高林　哲
奈良先端科学技術大学院大学
satoru-t@is.aist-nara.ac.jp

文書の山

我々は文書の山に囲まれて暮している．中でも電子メールは身近な存在である．

とあるシンポジウムでこんな出来事があった．討論の話題がIT（情報技術）革命に及んだときに，年輩の先生が次のような質問を投げかけた．「現時点でさえ，私は電子メールの処理に困っているのに，これ以上IT革命が進んだらどうなるのか」

この質問に対し，司会者はすかさず「すでにIT革命に乗り遅れてしまっている先生からの，大変いい質問です」と応え，場内は大いに沸いた．おそらく自分たちも困っているからこそ，多くの人が笑ってしまったのではないかと思う．

かくいう筆者も電子メールの処理に苦労している人間の1人である．筆者のメールボックスには約5万通，計200MBのメールが溜まっている．ほとんどのメールは後から参照することはないが，まれに参照したくなるときがある．たとえば，この問題の解決策は以前に誰かがメールで報告していたはずだ，と思い出したときなどである．

そこで，メールの山を検索する必要に迫られるのだが，単純な文字列検索のプログラムでは歯が立たない．試しに，手元の計算機でUNIXのgrepコマンドを使って筆者のメールボックスを検索したところ，実に12分もかかった．これでは短気な筆者などはとても我慢できない．

一方，高速な全文検索システムを使えば，同様のことが一瞬でできる．メールの数が倍々に増えていっても，検索に要する時間はほとんど変わらない．実際，筆者は大量のメールを一瞬で検索できる環境を構築している．

メールのほかにも，インターネットから集めてきたニュース記事やソフトウェアのマニュアルなど，知らず知らずのうちに文書は溜まっていく．気をつけていないと，文書の山に生き埋めになってしまいかねない．

Namazuで文書の山に立ち向かう

増え続ける文書の山を制するには，高速な全文検索システムがもはや不可欠である．全文検索システムというと，ODIN<http://odin.ingrid.org/>やGoogle<http://www.google.com/>といったWWWの検索エンジンが有名であるが，ここで必要とされているのはそのような巨大なシステムではなく，個人で使うための小規模なものである．Namazu[1]はまさにそのような用途のために作られている（図-1）．

インストール

Namazuは<http://www.namazu.org/>から入手できる．GPL2（GNU GENERAL PUBLIC LICENSE Version 2）に従ったフリーソフトウェアである．UNIX版とWindows版があるが，ここではUNIX版を元に説明する．

Namazuをインストールするためには，あらかじめ次のソフトウェアをインストールしておく必要がある．

- Perl バージョン5.004以降 <http://www.perl.com/>
- nkf バージョン1.71以降
  <ftp://ftp.ie.u-ryukyu.ac.jp/pub/software/kono/nkf171.shar>
  nkf 1.71付属のPerlモジュールNKFをインストールしておくとNamazuの動作が高速になる
- 茶筌 <http://cl.aist-nara.ac.jp/lab/nlt/chasen.html>またはKAKASI <http://kakasi.namazu.org/>

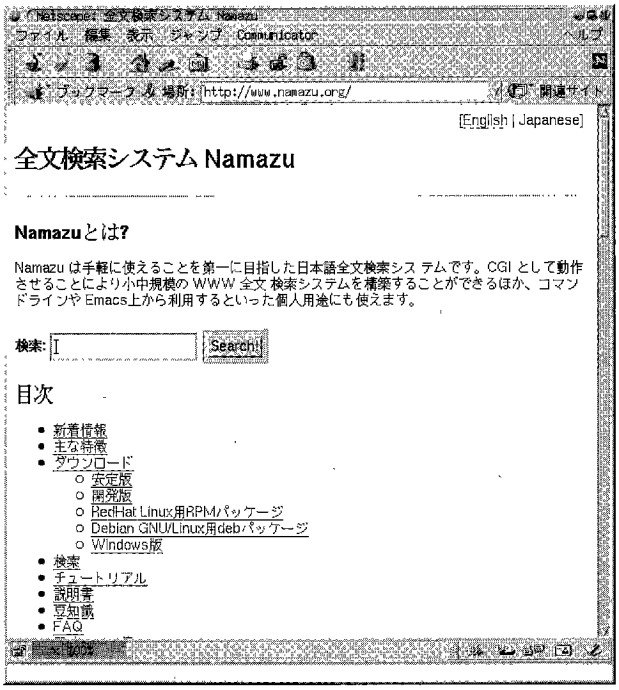

図-1　NamazuのWebサイト

```
% mknmz~/Mail/work
9個のファイルがインデックス作成の対象として見つかりました
1/9 - /home/foobar/Mail/work/1 [message/rfc822]
2/9 - /home/foobar/Mail/work/2 [message/rfc822]
3/9 - /home/foobar/Mail/work/3 [message/rfc822]
4/9 - /home/foobar/Mail/work/4 [message/rfc822]
5/9 - /home/foobar/Mail/work/5 [message/rfc822]
6/9 - /home/foobar/Mail/work/6 [message/rfc822]
7/9 - /home/foobar/Mail/work/7 [message/rfc822]
8/9 - /home/foobar/Mail/work/8 [message/rfc822]
9/9 - /home/foobar/Mail/work/9 [message/rfc822]
インデックスを書き出しています...
［基本］
日付:                  Sun Aug 27 10:41:33 2000
追加された文書の数: 9
サイズ（bytes）:       29,348
合計の文書数:          9
追加キーワード数:      1,233
合計キーワード数:      1,233
分かち書き:            module_kakasi -ieuc -oeuc -w
経過時間（秒）:        10
ファイル／秒:          0.90
システム:              linux
Perl:                  5.006
Namazu:                2.0.4
```

図-2　インデックス作成の過程

NMZ.body	NMZ.field.to	NMZ.result.normal
NMZ.body.ja	NMZ.field.to.i	NMZ.result.normal.ja
NMZ.field.date	NMZ.field.uri	NMZ.result.short
NMZ.field.date.i	NMZ.field.uri.i	NMZ.result.short.ja
NMZ.field.from	NMZ.foot	NMZ.slog
NMZ.field.from.i	NMZ.foot.ja	NMZ.status
NMZ.field.message-id	NMZ.head	NMZ.t
NMZ.field.message-id.i	NMZ.head.ja	NMZ.tips
NMZ.field.size	NMZ.i	NMZ.tips.ja
NMZ.field.size.i	NMZ.ii	NMZ.version
NMZ.field.subject	NMZ.log	NMZ.w
NMZ.field.subject.i	NMZ.p	NMZ.wi
NMZ.field.summary	NMZ.pi	
NMZ.field.summary.i	NMZ.r	

図-3　インデックスを構成するファイル

茶筌，KAKASIには，それぞれPerlモジュールText::ChaSen, Text::KAKASIが用意されている．Perlモジュールをインストールしておくとnamazuの動作が高速になる

Namazuのインストール方法は一般的なUNIXのフリーソフトウェアと同様である．ソースコードnamazu-2.0.x.tar.gzを入手して，次のように実行すればよい．

```
% gzip -dc namazu-2.0.x.tar.gz | tar xvf -
% cd namazu-2.0.x
% ./configure
% make
% su
Password:（rootのパスワードを入力）
# make install
```

詳しいインストール方法はNamazuに付属するチュートリアル（tutorial.html）を参照していただきたい．

インデックスの作成

Namazuを使って文書を検索するには，まずインデックスを作成する必要がある．インデックスとは，元の文書の情報を検索しやすい形で格納したデータ構造である．これについては後ほど詳しく説明する．

インデックスの作成はmknmzコマンドで行う．たとえば，~/Mail/work以下に保存されているメールを対象とするなら，コマンドラインから

```
% mknmz~/Mail/work
```

と実行する．処理が進むにつれて，図-2のようなメッセージが出力される．この例ではたった9個の文書しか対象としていないが，実際には数千，数万の単位の文書に対してもインデックスを作成できる．

mknmzの実行が終わると，mknmzを実行したディレクトリにNMZ.*ファイルが作られる．Namazuでは，これらのファイルをまとめてインデックスと呼ぶ（図-3）．

インデックスの大きさは，対象とする文書の量と性

```
/
└home
  └foobar
    └Namazu
      ├mydocs   ←自分の書いた文書用のインデックス
      ├manual   ←ソフトウェアのマニュアル用の〃
      └Mail     ←メール用の階層
        ├work   ←work メーリングリスト用の〃
        └staff  ←staffメーリングリスト用の〃
```

図-4　インデックスの管理（ディレクトリ構造）

```
% namazu 'コンパイル' ~/Namazu/Mail/work
検索結果

参考ヒット数: [コンパイル: 1]

検索式にマッチする1個の文書が見つかりました.

1. iconv（3）: boundary condition（スコア: 1）
著者: Satoru Takabayashi <satoru-t@is.aist-nara.ac.jp>
日付: Mon, 21 Aug 2000 10:17:37 +0900
 以下のプログラムをコンパイルして実行すると，OSによってiconv の挙動が異なります．Linux（glibc 2.1.3）の挙動はおかしくないですか？ これは既知の問題でしょうか．この問題を回避する方法はありますか?－
 /home/foobar/Mail/work/3（1,954 bytes）

現在のリスト:1 - 1
```

図-5　コマンドライン上での検索結果

質によって決まる．約2,000通，合計7MBのメールを対象に作成したインデックスは約4MBになった．多くの場合，インデックスは対象文書よりも小さくなるが，文書の量が少ないと，インデックスの方が大きくなってしまうことがある．

1つのディレクトリに対して複数のインデックスは置けないので，新しいインデックスを作るときにはディレクトリを新規に作成する必要がある．ホームディレクトリにNamazuというディレクトリを作り，その下にインデックス用のディレクトリをまとめるとよい（図-4）．mknmzに -Oオプションを指定すれば，インデックスの出力先のディレクトリを指定できる．

さきほど作成したインデックスを~/Namazu/Mail/workに移動するには，mknmz-p~/Namazu/Mail/work;mv NMZ.*~/Namazu/Mail/workと実行する．

インデックスの更新

上の例では~/Mail/work以下に保存されているメールを対象にインデックスを作成した．メールを利用しているうちに，新しいメールが溜まってきたときは，

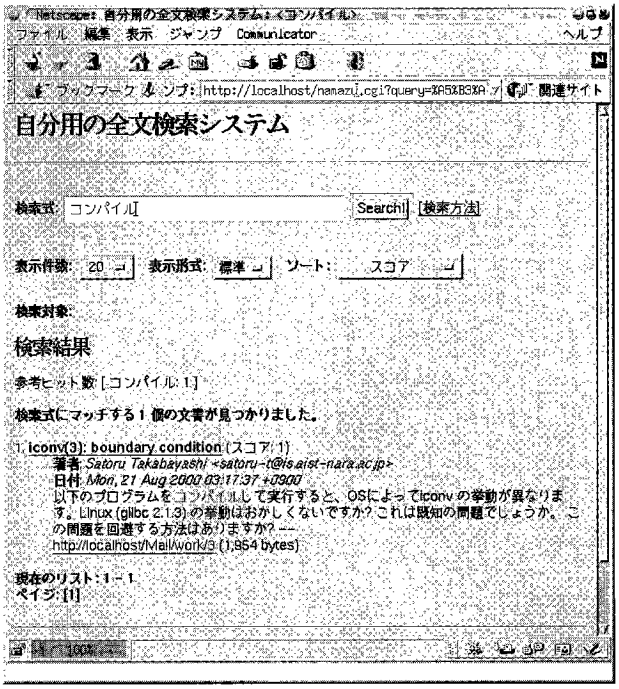

図-6　Webブラウザ上での検索結果

次のように実行してインデックスを更新する．インデックスは~/Namazu/Mail/workに置かれているものとする．

```
% mknmz --update~/Namazu/Mail/work
```

更新の際には新しく追加されたファイルを対象とするため，短時間で処理できる．もし削除されたファイルや更新されたファイルがあればそれらも反映される．

検索

検索はnamazuコマンドで行う．さきほど作成したインデックスに対し，キーワード「コンパイル」で検索するにはコマンドラインから次のように実行する．

```
% namazu 'コンパイル' ~/Namazu/Mail/work
```

検索結果は図-5のように表示される．Webサーバと連携すれば図-6のようにブラウザから検索できる．

上の例では単純に1つのキーワードを指定しただけである．Namazuには，複数のキーワードの指定，正規表現によるキーワードの指定といった検索方法も用意されている．詳しい使い方は付属するマニュアル（manual.html）を参照して欲しい．

単語	その単語を含む文書のID
algorithm	1, 4, 6, 7
computer	1, 3
information	3, 4
java	4, 7, 8
lisp	7
processing	1, 3, 4, 8
scheme	6, 7

図-7　転置ファイル

全文検索システムの仕組み

全文検索システムとは，文書全体に含まれる文字列を検索するシステムである．ここではNamazuで採用している手法を例にとって全文検索システムの仕組みを解説する．

転置ファイル

大量の文書を高速に全文検索するためには，検索対象の文書を元にインデックスを作成する必要がある．

インデックスの作成方法には多くの手法があるが，最も一般的なものは，単語と，その単語を含む文書のIDとの対応表を作る方法である．たとえば，computerという単語は文書1と文書3に含まれ，informationという単語は文書3と文書4に含まれるといった対応表を作り，単語のアルファベット順に並べる．これを転置ファイル（inverted file）と呼ぶ．文書IDの代わりにページ数を見立てると，ちょうど書籍の索引と同じ形になる（図-7）．

検索時には，転置ファイルの単語の表を2分探索することにより，与えられたキーワードを含む文書が高速に検索できる．

文書IDの処理

複数のキーワードを指定して検索するときは，文書IDの集合に対して集合演算を適用する．たとえば，computerとinformationの両方を含む文書を検索（AND検索）するときは，それぞれの単語で検索して得られる文書IDの集合 {1, 3}，{3, 4} の積集合をとると，{3} という結果が得られる．

文書IDはソート済の状態で転置ファイルに格納されているため，集合演算は高速に行える．また，ソート済という特性を活かして文書IDのデータを圧縮することもできる．Namazuでは文書IDを差分で記録してBER圧縮をかけている．BER圧縮はPerlのpack関数で提供されている．より高度な圧縮手法としてはElias-γ法などがある[2]．

フレーズの検索

上の例では，単語と文書IDを対応させて転置ファイルを作成していたが，これでは "information processing" のようなフレーズが正確に検索できない．informationとprocessingをAND検索しただけでは，"processing of information is ..." といった文を含む文書までもヒットしてしまう．

フレーズの検索を実現するためには，文書IDとともに単語の出現位置を記録すればよい．検索時にはAND検索で積集合をとるとともに，出現位置の照合を行う．

単語の出現位置を記録すると，その分，転置ファイルが大きくなる．Namazuでは，ファイルの大きさを抑えるために，ハッシュを用いた手法でフレーズ検索を実現しているが，今にして思えば，たいしてサイズの節約になっていない割に，検索の精度を落とすという弊害の方が大きいようである．将来のバージョンでは出現位置を記録する手法に直すつもりでいる．

ストップワード

転置ファイルの作成時に，キーワードにむかない単語を除外する戦略がある．a, the, of, toなどといった頻出する単語（ストップワードと呼ばれる）を除外すれば，転置ファイルを小さくすることができる．しかし，弊害として，"to be or not to be" のようなフレーズが検索できなくなる．Namazuでは弊害の方を重視して，ストップワードの除外は行っていない．

分かち書き

転置インデックスの手法を日本語の文書に対して適用しようとすると大きな壁につきあたる．英語の場合は語と語の区切りが明白であるため，単語の表を容易に作れるが，日本語の場合は区切りに曖昧性があるため同じようにはいかない．

そこで，日本語の区切りの曖昧性を解消し，適切に分かち書きするためのプログラムが必要となる．NamazuではKAKASIまたは茶筌を利用している．KAKASIは高橋裕信氏によって開発された，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プログラムである．分かち書き処理への対応は馬場肇氏によって行われた．茶筌については本特集の記事を参照していただきたい．

茶筌で「Namazu は手軽に使える全文検索システムである」を分かち書きすると，「Namazu・は・手軽・に・使える・全文・検索・システム・で・ある」のようになる．

茶筌・KAKASIによる分かち書きの出力が不適切であった場合は，本来は見つかるはずの文書が見つからないという状況が起きる．一方，分かち書きを行わない手法としてはSuffix Arrayを用いるものやシグネチャを用いるものなどがある．文献2）では転置ファイルの作成方法も含め，それぞれの手法が詳しく解説されている．

ランキング

検索結果は，有用な文書が上位となるように表示するのが望ましい．有用な文書とは利用者が与えた検索質問（query）に対して最も適合する文書のことをいう．

検索システムは，検索質問と検索にヒットした文書との適合度を計算し，それに応じてランキングを行う．ランキングの単純な手法としては，検索に使われたキーワードをたくさん含む文書ほど優先する，というものがある．HTMLなどの構造を持った文書に対しては，タイトルや見出しに含まれる単語に重みをつけることが多い．

複数のキーワードが指定されたときは，ありふれた単語の重みを低くして全体の適合度を計算する．たとえば，seminumerical, programming, easyと3つの単語を指定して検索したときに，それぞれの単語を等価に扱えば，easyをたくさん含んだ文書が検索結果の上位にきてしまうだろう．この例ではseminumericalという希少なキーワードを優先した方がよい結果が得られる．

ありふれたキーワードに低い重みを，希少なものに高い重みをつける伝統的な手法としてtf idf法がある．キーワードwが与えられたときの文書dのtf（term frequency）とidf（inverted document frequency）の値は次のように計算する．

tf ＝文書dの中のキーワードwの出現頻度
idf ＝log（全文書数/キーワードwを含む文書の数）

tf idfはこれらの積である．

Namazuでは，検索質問に含まれる各キーワードごとにtf idfの値を求め，それらを単純に足し合わせることによって適合度を計算しているが，一般的にはベクトル空間モデルによって計算されることが多い．ランキング手法については文献3）に詳しい．

Namazuの生い立ち

Namazuは1997年の夏に生まれた．当時，大学3年だった筆者は，1つの問題を抱えていた．それは，自分用に作った全文検索システムが遅くて使いものにならない，ということであった．

筆者の手元には，WWWから集めた約1,000ファイルの文書があったのだが，それらを検索するのに30秒もかかった．世界中のWebサイトを一瞬で検索できる検索エンジンが世の中には存在するというのに，たかだか1,000ファイルの検索に30秒もかかってしまうとは情けない話である．

そこで，全文検索システムの仕組みについて調べてみると，簡単なシステムならすぐに作れそうだと気づいた．さっそく開発に着手し，1週間後には動くものができた．

その後は，メーリングリストでの宣伝，Webサイトの開設，一般公開と，とんとん拍子で進んだ．利用者が増えるにつれて完成度が高まり，いつの間にか広く使われるフリーソフトウェアになっていた．バージョン2.0からはインターネット上での共同開発が進められている．

名前の由来

Namazuという名前の由来は特にない．ふと思い浮かんだだけの名前である．知人からは「世界を揺るがすからNamazuなんだ，と言っておきなさい」と言われているが，どうもこれは大仰な気がするし，そもそもナマズは地震に反応するだけ（予知するという説もある）であって，地震を起こすものではない．

なぜ普及したのか

文献4）に掲載されている，Namazuを採用したWebサイトの一覧を見ると，驚くほど多くの組織でNamazuが利用されていることが分かる．なぜこれほど普及したのか自分でも不思議である．考えられる理由は

こんなところだろうか．

- フリーソフトウェアである
- Webサイトの全文検索システムに使える
- 個人の用途に使える
- 比較的，簡単に使える
- 競合するソフトウェアが少ない

振り返ってみると，初期のバージョンの完成度は実にひどかった．しかし，なんとか「動いて」「使える」しろものではあったと思う．初めから完成されたソフトウェアを開発しようとするより，要求の半分程度を実現するものをひとまず公開し，それから徐々に改良して共同開発の体制に持っていく方が成功するようである．この事実は「デザインの『悪い方がよい』原則」として知られている[5]．以下にその主旨を引用する．

最初に「正しい」方法をとることはしばしば望ましくないということである．とりあえず「正しい」ことの半分はできるものを作り，ウイルスのように広める方がよい．いったん人々がそれに騙されれば，「正しい」ことの90％までできるように改善が行われるだろう．

Namazuの行く末

Namazuは枯れた技術だけを利用して作られた素朴な全文検索システムである．情報検索の分野では，自然言語による検索や概念検索といった高度な技術の研究が行われているが，Namazuではそれらの事情をほとんど考慮に入れていない．それでも結果的にNamazuがこれだけ普及したところをみると，実用的で広く使われるソフトウェアを作るには，枯れた技術だけでも十分ということらしい．マイクロカーネルが盛んに研究された時代にあって，伝統的なモノリシックカーネルのLinuxが成功を収めたのも，その1つの例であろう．

Namazuがもっと現代的で高度な検索技術を採用すべきかは悩めるところである．おそらくNamazuに求められているのは，高度な検索技術よりも，手軽に日常的に使えるソフトウェアであることだろう．その意味では，使い勝手のよさを追求するべきなのだが，今後は筆者の勉強も兼ねて，情報検索の現代的な手法を少しずつ取り入れていこうと思っている．

しかし，新しい手法を取り入れようにも，現在のNamazuは拡張が難しい状態にある．3年前にNamazuを作り始めたときは，とにかく動けばいい，という一心でプログラミングしていた．そして，機能が増えるにつれてプログラムの複雑性が増大し，今では手がつけられない状態になっている．

そこで，筆者は現在Namazuの再設計に取り組んでいる．Namazuの開発を通じて得られたプログラミングの教訓はたくさんあるが，最も大きいのは，きちんと設計せよ，というものだ．初期の貧弱な設計のおかげで，その後，ずっと苦労するはめになった．

新しい設計では，保守がしやすく，自由自在に拡張できるソフトウェアを目指している．また，使いやすいライブラリの提供も大きな目標である．全文検索の機能をいろいろなアプリケーションに組み込めるようにしていきたい．

Namazuは研究プロジェクトではなく，実用を目指すフリーソフトウェアである．その開発が学問として新たな知見をもたらすことはないかもしれないが，よりよい計算機環境の追求のために今後も開発を続けていきたい．ナマズの寿命は60年だそうだが，Namazuはこの先何年，使われているだろうか？

参考文献

1) NamazuのWebサイト <http://www.namazu.org/>
2) Baeza-Yates, R. and Rieiro-Neto, B.: Modern Information Retrieval, Addison Wesley (1999).
3) 原田昌紀: サーチエンジンにおける検索結果のランキング, bit, Vol.32, No.8, pp.8-14 (2000).
4) 馬場　肇: 日本語全文検索エンジンソフトウェアのリスト <http://www.kusastro.kyoto-u.ac.jp/~baba/wais/other-system.html>
5) Gabriel, R. P. (持橋大地訳): デザインの「悪い方がよい」原則 <http://cl.aist-nara.ac.jp/~daiti-m/text/worse-is-better-ja.html>

(平成12年9月28日受付)