情 報 シ ス テ ム 48－7
（1994．3．15）

複数ハッシュふるい分け法の日本語情報システムへの応用

中本賢一　　山本毅雄　　長谷部紀元
図書館情報大学

概要

複数ハッシュふるい分けを用いた索引語切り出し機能及び検索質問の単語分解機能をWAIS(Wide Area Information Server)に付加し、べた書き日本語テキストの自然語による検索を可能にした。複数ハッシュふるい分けでは、辞書から作成するハッシュ表を使い、あるキーがその辞書中に存在するかどうかを調べる。ハッシュ表にはキーの有無を示す１ビットのフラグだけが立っているので、もとの辞書より必要な記憶容量がはるかに小さい。これを利用し日本語テキストと質問文の両方から同じ辞書、同じ方法でキーを切り出すことができ、必要に応じた辞書を使用できる。また文章中の多くの単語の組合せがキーになるため、送り仮名等の表記のゆれにある程度対応できる。

Screening by multiple open hashing for a Japanese information system

Ken'iti Nakamoto, Takeo Yamamoto and Kigen Hasebe
University of Library and Information Science

Abstract

A Japanese language retrieval capability is incorporated in WAIS (Wide Area Information Server) through the use of multiple-hash screening technique. Here, the hash table is constructed by hashing each dictionary entry with a number of mutually independent hash functions. Whether the search key fits with a dictionary entry is judged solely from the hash table, without accessing the original dictionary. As the hash table consists only of single-bit flags, it is much smaller than the original dictionary. Consequently, it is possible to process both the Japanese text data and the query string based on the same dictionary, which can be changed from database to database. As many combinations of possible words in the text may be indexed, some variations in kana/kanji expressions may be tolerated in the retrieval.

1 はじめに

インターネット上では WAIS (Wide Area Information Server), Gopher, WWW など多くの検索システムが公開され、これを使い世界中のテキストデータを広く検索することが可能になってきた [1, 2]。しかし、これで検索できる日本語テキストデータはまだごく少ない。これは欧米語と異なり、日本語では単語の切れ目がはっきりせず、切り分けが困難なためである。ここでは辞書そのものより記憶容量の小さいハッシュ表を用いた、複数ハッシュふるい分けによって、日本語テキストデータの切り分けをおこなう方法 [3] を用い、WAIS に自然語による日本語検索機能を付与したシステム、MWAIS を試作した。単語の切り分けに日本語形態素解析システム JUMAN [4] を用いた WAIS+ [5] があるが、MWAIS ではデータベースごとに異なる辞書 (ハッシュ表) を用いることができること、可能な全ての切り分けを利用したもれの少ない検索が出来ること等が特徴である。

2 WAIS の日本語検索機能

2.1 WAIS について

WAIS [6] はインターネットによって結ばれたデータベースを世界中で利用するための、いわばデータベースのデータベースである。WAIS はクライアント・サーバ方式で機能する。

WAIS を利用するユーザは、まず世界中に多数あるデータベースサーバの中から、検索すべきサーバを選択するため、WAIS クライアントから WAIS 情報源ディレクトリ (WAIS Directory of sources) を持つサーバにアクセスして検索する。

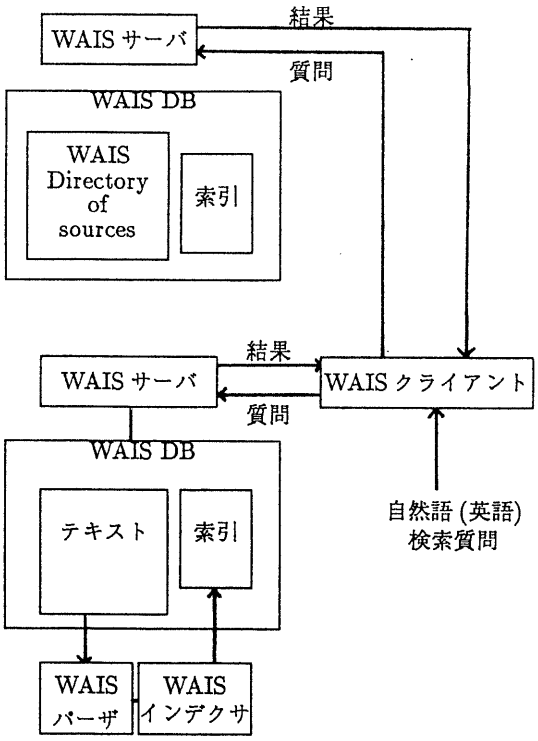

図 1: WAIS

このサーバは、最適なデータベースサーバから順に重みをつけたデータベースのリストを返す。ユーザはそのリストの中から、使用するデータベースサーバを選択し、これに対する WAIS クライアントに質問文を入力する。WAIS クライアントはこれを変換してサーバに送り、検索の結果としてサーバから順位のついたテキストのリストを受け取る。ユーザはこの中から必要なテキストを選択し見ることができる。

サーバでは前もって WAIS パーザ及び、WAIS インデクサによってテキストから索引キーを切り出し、その結果から索引を作成する (図 1)。

本研究では WAIS を図 2 のように改

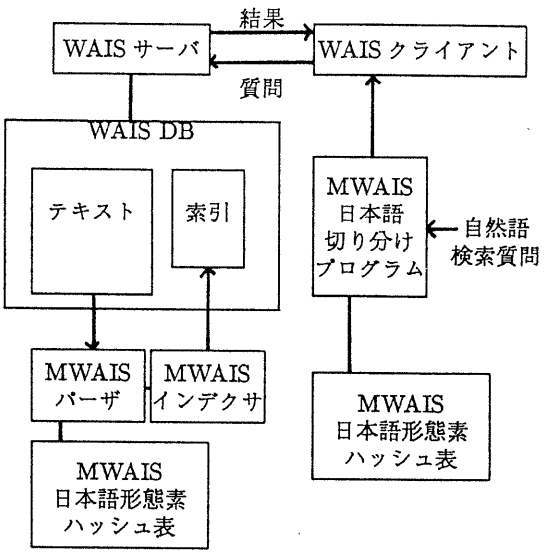

図 2: WAIS への日本語検索機能の付与

造し、日本語による検索をおこなう機能を付与した。すなわち、WAIS パーザ、及び WAIS クライアントに同じハッシュ表を使用する日本語切り分けプログラムを組み込み、日本語テキストデータの検索を可能にした。

3 複数ハッシュふるい分けについて

ふるい分けとはあるキーが辞書中に存在するかどうかを調べることである。複数ハッシュふるい分け [7, 8, 9] では、辞書からあらかじめ作成しておいたハッシュ表を使いふるい分けをおこなう。

ハッシュ表を作るには、辞書中のキーをハッシュ関数 (付録) によって複数個のハッシュアドレスに変換し、対応するアドレスを持つハッシュ表のエントリに 1 ビットのフラグを立てる。この操作を辞書中のキー全てについておこないハッシュ表を作成する。ふるい分けをおこなうときには、対象となるキーから、ハッシュ表を作成したときと同じ操作で、複数個のハッシュアドレスを生成し、これらに対応するエントリの内容を調べる。全てのエントリに 1 ビットのフラグが立っていれば、そのキーは辞書中に存在すると判定し、少なくとも一つのエントリにフラグが立っていなければ、そのキーは辞書中に存在しないと判定する (図 3)。

この方法では辞書にアクセスせずに高速でふるい分けが可能であるが、異なるキーから同じハッシュアドレスが生成される (衝突が起こる) と、実際は辞書中に存在しないキーを、辞書中に存在すると判定してしまう。この確率を充分小さくするために、複数個のハッシュ関数を使う。本研究ではさらに、文字列の畳み込みにおける衝突を避けるため、付録に示す方法を開発した [3]。

	h_0	h_1	h_2
0		1	
1	1		
2			1
3		1	
4	1		
5			1

h_0(“ABC”) = 4, h_1(“ABC”) = 0,
h_2(“ABC”) = 2
“ABC” は辞書中に存在する。

h_0(“EFG”) = 2, h_1(“EFG”) = 3,
h_2(“EFG”) = 3
“EFG” は辞書中に存在しない。

図 3: 複数ハッシュふるい分け

- 切り分けの対象となる文字列：

「大学図書館」

- ふるい分けされる部分文字列：

大, 大学, 大学, 大学図, 大学図書, 大学図書館
学, 学図, 学図書, 学図書館
図, 図書, 図書館
書, 書館
館

図 4: ふるい分けの対象

ハッシュ表の大きさは辞書そのものの大きさより小さくなる。この特徴を生かし、サーバから、クライアントにハッシュ表を転送して、質問文ともとのテキストデータを同じ辞書で切り分けることができる。

3.1 文章の切り分け

MWAIS では現在、句読点などの区切り記号で区切られた文字列を対象とし、これに含まれる全ての部分文字列についてふるい分けをおこなっている (図 4)。

ここで得られる辞書中に存在する部分文字列 (形態素候補) について、もとの文字列全体を構成するのに必要な形態素候補だけを残すために、その前後につながる形態素候補があるかどうかを調べる。前後いずれか一方でもつながらない場合には、この形態素候補を捨てる。以上の操作で残った、形態素候補は、構文的、意味的情報を無視し、辞書中に存在するかどうかだけを基準に選んだ全ての形態素の可能な組合せを含んでいる (図 5)。

- 切り分けの対象となる文章：

「大学図書館における相互目録の活用状況について述べる」

- 残る形態素候補：

大, 大学, 学, 図, 図書, 図書館, 書, 館, に, にお, お, おけ, おける, け, ける, る, 相互, 目, 目録, 録, の, 活, 活用, 用, 状況, に, につ, につい, について, つ, つい, い, いて, て, 述, 述べ, 述べる, べ, る

図 5: キーの抽出

もとの文字列の中に、辞書中に存在しない “未定義語” がある場合、上の方法で切り分けが不可能な場合がある。このときは、もとの文字列の先頭と最後尾から、それぞれつながる形態素候補を調べ、その二つの形態素候補に挟まれる部分文字列を未定義語の一部とした。そして未定義語の一部の字種と、その前後の文字の字種を調べ、同じ字種の続く範囲を未定義語とした。こうして抽出される未定義語は索引や検索に使用することもできる。

実験に使用した辞書は ICOT フリーソフトウェア「形態素辞書」[10] のエントリに、同辞書に含まれる動詞、形容詞の語幹及び活用語尾から作成した活用形を加えたもので、約 23 万件の語からなる。使用するハッシュ表の大きさは約 1MB である。

3.2 表記のゆれへの対応

自然語による検索で生じる問題の一つに、送り仮名の違いによる表記のゆれが

ある。例えば「割り込み」「割込み」「割り込」「割込」などの場合がそれである。この場合、複数ハッシュふるい分けによって文章の切り分けをおこなうと、得られるキーはそれぞれ以下のようになる。

- 「割り込み」
  割, 割り, 割り込, 割り込み, り, 込, 込み, み
- 「割込み」
  割, 割込, 割込み, 込, 込み, み
- 「割り込」
  割, 割り, 割り込, り, 込
- 「割込」
  割, 割込, 込

この場合はどのような送り仮名の付け方でも、重みに差はあっても検索することが可能である。

その他、長音の表記の違いによるゆれがある。「チャリティコンサート」と、「チャリティーコンサート」等がその例である。現在の辞書には「チャリティ」があり、「チャリティー」が含まれていない。この場合、抽出されるキーは以下のようになる。

- 「チャリティコンサート」
  チャリティ, コンサート
- 「チャリティーコンサート」
  チャリティ, ー, コンサート,

なおここで、「チャリティーコンサート」から抽出されるキーの「ー」は未定義語の一部である。そして未定義語をキーとするとさらに「チャリティーコンサート」がキーとなる。

MWAIS では、検索対象のテキストと質問文から、同じ辞書、同じ方法を使い、全ての可能な切り分けをおこなうことから、このようなゆれに対応することができる。同様にテキストと質問文に同じ未定義語が出現した場合にも、未定義語を索引キー、検索キーとして扱うことによって、検索が可能になる。

4 まとめ

試作した MWAIS の実行例を図 6 に示す。

計算機上に大量のテキストデータが分散して存在しているとき、これを効果的に検索する上で、自然語による検索を実現することは重要である。複数ハッシュふるい分けを用いた MWAIS では、同じ辞書、同じ切り分け方を使って、検索対象であるテキスト、質問文を切り分けることによって、検索もれの少ない検索をおこなうことが出来る。

MWAIS では、ハッシュ表が辞書そのものよりもはるかに小さくなることを利用して、サーバごと、あるいはデータベースごとにでも、辞書を変えることが可能なので、例えば国文学のテキストと物理のテキストで異なる辞書を用いて索引を作成し、検索の際にもそれぞれ適した辞書を用いて質問文を切り分けることが可能となるであろう。また比較的メモリの小さいパソコンでも実現が可能であり、より広範な利用が期待できる。

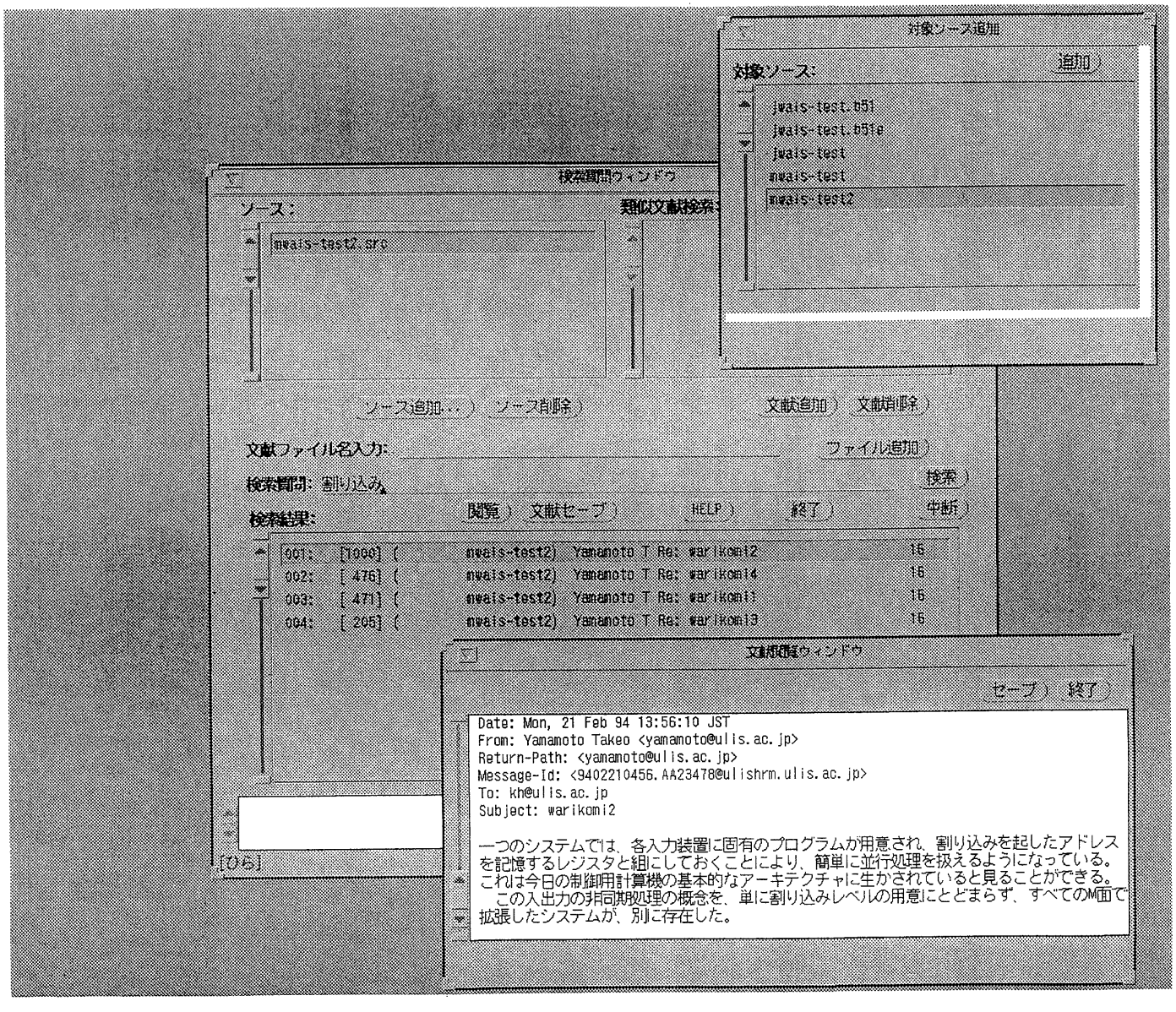
対象ソース追加
対象ソース:
追加)
jwais-test.b51
jwais-test.b51e
jwais-test
mwais-test
mwais-test2
検索質問ウィンドウ
ソース:
類似文献検索
mwais-test2.src
ソース追加..)
ソース削除)
文献追加)
文献削除)
文献ファイル名入力:
ファイル追加)
検索質問: 割り込み
検索)
検索結果:
閲覧)
文献セーブ)
HELP)
終了)
中断)
001: [1000] (mwais-test2) Yamamoto T Re: warikomi2 15
002: [476] (mwais-test2) Yamamoto T Re: warikomi4 15
003: [471] (mwais-test2) Yamamoto T Re: warikomi1 15
004: [205] (mwais-test2) Yamamoto T Re: warikomi3 15
文献閲覧ウィンドウ
セーブ)
終了)
Date: Mon, 21 Feb 94 13:56:10 JST
From: Yamamoto Takeo <yamamoto@ulis.ac.jp>
Return-Path: <yamamoto@ulis.ac.jp>
Message-Id: <9402210456.AA23478@ulishrm.ulis.ac.jp>
To: kh@ulis.ac.jp
Subject: warikomi2
一つのシステムでは、各入力装置に固有のプログラムが用意され、割り込みを起したアドレスを記憶するレジスタと組にしておくことにより、簡単に並行処理を扱えるようになっている。これは今日の制御用計算機の基本的なアーキテクチャに生かされていると見ることができる。
この入出力の非同期処理の概念を、単に割り込みレベルの用意にとどまらず、すべてのM面で拡張したシステムが、別に存在した。
[ひら]

図 6　MWAIS 実行例

謝辞

卒業研究で MWAIS を実現に協力された藤原誠治君に感謝します。

参考文献

[1] Ed Krol. *THE WHOLE INTERNET : USER'S GUIDE & CATALOG*. O'REILLY & Associates, 1992.

[2] Mark Gibbs and Richard Smith. *NAVIGATING THE INTERNET*. Sams Publishing, 1993.

[3] 中本賢一, 山本毅雄. 日本語形態素解析への複数ハッシュふるい分けの応用. 第 *47* 回 *(*平成 *5* 年度後期*)* 全国大会 講演論文集 *(3)*, pp. 157–158, 1993.

[4] 松本裕治, 黒橋禎夫, 妙木裕, 長尾真. 日本語形態素解析システム *JUMAN* 使用説明書 *Version 1.0*. 京都大学工学部 長尾研究室, 奈良先端技術大学院 大学 松本研究室, 1993.

[5] 渡辺洋一. 日本語 *WAIS (wais+ 0.3.3 release)* について, 1993.7. 私信.

[6] WAIS Incorporated. *WAIS Server WAIS Workstation WAIS Forwarder for UNIX*, 1993.

[7] M. Douglas McIlroy. Development of a spelling list. *IEEE Transactions on Communications*, Vol. COM-30, No. 1, pp. 91–99, 1982.

[8] 渋谷政昭, 山本毅雄. データ管理算法. 岩波講座情報科学 11. 岩波書店, 東京, 1983.

[9] Craig Stanfill and Brewster Kahle. Parallel free-text search on the connection machine system. *Communication of the ACM*, Vol. 29, No. 12, pp. 1229–1239, 1986.

[10] Institute for New Generation Computer Technology. *ICOT* フリーソフトウェア *No.33* 形態素辞書 *Version 1.1*, 1993.

付録・ハッシュ関数について

日本語文字列のような長いキーをハッシュアドレスに変換する操作であるハッシュ関数の計算は、次の二つの段階に分けて考えられる。

1. $T(Z) = K$　文字列 Z を比較的長いビット長の値であるハッシュキー K に変換する。この操作を畳み込みと呼ぶ。

2. $h(K) = A$　ハッシュキー K をハッシュアドレス A に変換する。

複数ハッシュふるい分けでは、文字列 Z を $T(Z)$ によりハッシュキーに変換した後、複数個の $h_i(K)$ を用いて複数個のハッシュアドレスを生成する。そのため $T(Z)$ の計算によって異なるキーが同じ値に変換されてしまうと、その後にいくつの $h_i(K)$ を施したところで衝突を避けることはできない。

そこで文字列の畳み込みの方法について充分検討することが必要になる。本研究では、文字に与えられる文字コードを出発値とする疑似乱数列を作成し、文字列におけるその文字の出現位置に応じて、同じ文字でも異なる値をその文字のコードとし、その文字列を構成する全ての文字のコードを決定し、それらの排他的論理和を取る方法を採用した。使用した疑似乱数列生成関数は

$$r_i = (a \times r_{i-1}) \bmod m$$

であり、$a = 16807$, $m = 2^{31} - 1$ である。r_i は i 番目の疑似乱数であり、出発値となる文字コードを z としたとき

$$r_0 = (a \times z) \bmod m$$

である (図1)。

	r_0	r_1	.....	r_{13}	r_{14}
z_0			.....		
z_1			.....		
z_2			.....		

疑似乱数を用いた文字コード表

“ABC” を畳み込む時は、

$T($ “ABC” $) = f^0($“A”$) \oplus f^1($“B”$) \oplus f^2($“C”$)$

を計算する。$f^i(z)$ の値は全ての文字についてあらかじめ計算しておく。

図 1: 疑似乱数列による畳み込み

この方法を使い、ICOT フリーソフトウェア「形態素辞書」をもとに作成した約 23 万件のエントリからなる辞書について畳み込みをおこなったところ、発生した衝突の数は 26 件であった。

更に衝突を抑えるために、二つの異なる関数を使用して疑似乱数を生成し、一つのキーについて二つの疑似乱数列を使い、二つの値に畳み込んだ。使用した辞書中のキーについて、生成される二つのハッシュキーの組合せが同じ値になることはなかった。使用した疑似乱数列生成関数は先に挙げたものと、$a = 42024$ としたものの二つである。

ここでは二つの方法で畳み込みをおこない、生成される二つの値からそれぞれ 4 つ、全部で 8 つのハッシュアドレスを生成し複数ハッシュふるい分けをおこなう。