テキスト型ドイツ語データベースとデータ検索法について

鬱田 收[†]　高瀬 誠[††]

1　はじめに

本プロジェクトではこれまで、Wilhelm Dilthey 著作集を OCR システムにより読み取り、一定の書式を持つテキスト型データベースとする作業を行ってきた[1]。本稿では、ここで作成されたテキスト型データベースから、具体的なデータを取得するための検索法とそのインターフェースについて行った考察と、それに基づいて試作された検索システムについて報告する。

2　前提条件

2.1　前提となるデータとその構造

検索対象となるデータとその構造については、95 年の報告「OCR によるデータベース作成」で詳述したが、今一度その概略をふりかえると、次の通りとなる:

- ここで検索対象として利用されるのは、本プロジェクトでデータ化してきた Wilhelm Dilthey 著作集[2]からのテキストである。
- データは、awk や Perl で容易に扱えるよう複行レコード形式として記録されている。1 レコードが持つデータ・フィールドは次の通りである:
  1. ページ番号
  2. 当該テキスト行 (1 行 1 フィールド + フィールド区切り子 (改行 1 個)) 複数
  3. レコード区切り子 (連続する改行 2 個)
- データ中の欧文特殊文字はや、イタリック体などの組版上の特性などで必要と思われるものの記述は、TEX, LATEX での記述方式に従う。特にドイツ語に特有の文字は、ドイツ語用スタイル・ファイル german.sty により定義された記述方式に従う[3]。

† 文学部教授、E-Mail: Osamu.Kutsuwada@gakushuin.ac.jp
†† 学習院大学非常勤講師、E-Mail: Makoto.Takase@gakushuin.ac.jp

[1] 本プロジェクトは、計算機センター特別研究費により行なわれた。

[2] Dilthey, Wilhelm: Gesammelte Schriften, B. G. Teubner Verlagsgesellschaft, Stuttgart, Vandenhoeck & Ruprecht, Göttingen.

[3] テキスト型データの汎用性を重視したマークアップ言語としては SGML も挙げられる。これには

その他データ構造に関する詳細は、[2] を参照されたし。

2.2 前提となるユーザ像

一般にあるシステムを構築する場合、そのシステムを利用するユーザ像が明確になっていることで各種の作業が容易なものとなる。本プロジェクトでも、作成されるシステムを利用するユーザ像をある程度決めておき、その上でシステムを構築していくことにした。

さまざまな検討を行った結果、ここではシステムを利用するユーザ像として次のようなものを想定することとした:

1. 自分の利用している計算機環境の基本的操作法を周知していること

   具体的には以下の点を最低限満たしている必要があろう[4]:

   - エディタ、ワードプロセッサなどの各種ツールの基礎的かつ基本的操作法について理解している
   - ネットワークの基礎的基本的な利用方法について理解している (ネットワーク関連の各種ツールの基本的な利用方法を理解している)
   - 普段利用しているシステムについて不明な点・問題点などがあれば、自ら解答・解決策などを探す方法を周知している

2. 自分が普段利用している計算機環境とは別の環境に対しても、十分に対応できること

   ここで試作されるインターフェースや出力データは、すべてのユーザにとって普段利用している計算機環境とぴったり合致するものとは限らない。例えば、検索対象となるデータは、特定の計算機環境に依存しない形式で記述されている。検索の結果取得したデータも同一の形式で記述されているため、場合によってはユーザにその形式を自由に変換する能力が必要となる。

3. 未知の概念や機能についても自ら調査・実験を繰り返し、利用してみるだけの積極性があること

– 特定のシステムに依存していない「上位の」マークアップ言語である
– テキストの持つ構造を精密に記述できる
– パーサを通してその他の「下位の」マークアップ言語への移植もできる

などの特徴がある。

本プロジェクトでも開始当初はデータを SGML で記述することも検討された。しかし調査の結果、SGML はテキスト構造の記述には大変強力な機能を有しているものの、個別言語の記述に対しては極めて脆弱な機能しかなく、記述方式も TEX, LATEX に比べてはるかに繁雑で、可読性の確保は困難であることが判明した。

また、SGML のサブセットのような形で構成されている HTML でも、個別言語記述の方式は SGML での方式が踏襲されており、そこから推察すると、とりわけマルチバイト・コードとの親和性を確保することは少なくとも現段階では難しいと考えられる。

以上のような理由から、本プロジェクトでは SGML でのデータ記述は行わないことにした。しかし、SGML の持つ「システムに依存しないテキスト型データの記述」という基本理念は正しいものと考え、その基本理念を先行する形で実現した TEX, LATEX の記述方式を採用することとした。

なお SGML に関しては、[17] などを参照。

[4] ここで挙げられている条件は少なくとも本学の「初等情報処理」の授業で習得できるものと思われるので、無理な要求とは考えられない。

テキスト型データからの検索は、求めているデータが必ずしも容易に入手できるとは限らない。さまざまな試行錯誤が必要となることもある。

そうした場合に例えば、「正規表現を利用した検索」が極めて強力なものとなり得るが、その際、「正規表現」というものについて、たとえ知識がなくとも、どのようなものなのか、どのような特徴があるのか、どのように使うものなのか、その長所・短所は何か、といったことはある程度の調査や実験で比較的容易に知ることができる。

ユーザはそうした調査・実験を積極的に行い、システム等に関する理解を深め、当該ツールの持つ利用可能性を— 場合によっては設計者の意図を越える程度までに — 大きく拡げることが望ましい。

3 基本構想

3.1 概要

以上のような前提条件に基づくと、ここで試作されるテキスト型データからの検索用インターフェースに関する概要は次のようになろう:

Text 型データから検索

検索対象は当面、本プロジェクトで作成した W. Dilthey 著作集のテキスト型データとするが、他の同形式データからも検索が可能となるようにしておく必要があろう。

検索文字列には正規表現を指定

テキスト型データに対して各種の検索を行うためには、多様で柔軟な検索用文字列が使える必要がある。そうした検索文字列の中でも、「正規表現」は現在では最も強力なもので、各種の検索用ツールで広くサポートされている。ここで試作される検索用インターフェースでも、この正規表現をサポートすることでより柔軟で多様な検索ができるようになると考えられる[5]。

検索ツールとして Perl を利用

本プロジェクトでは当初から、検索ツールとして awk によるごく簡単なスクリプトを実験的に作成していた。しかし、今回検索用インターフェースを試作するに当たって、これを Perl によるスクリプトに変更した。その理由は以下の通りである:

- Perl は基本的にフリーソフトとして配布されているので、誰でも容易に入手できる。また現行のほとんどすべてのプラットフォーム上に移植されているため、特定のシステムに依存したス

[5] ただし、正規表現はサポートしているツールによって微妙に異なる部分もある。現時点でもっとも効率の良い検索を行うには、最適なツールを選択する必要もある。
その他、正規表現の詳細については、[4], [5], [9], [14], [16], [18], [27], [28] などを参照。

クリプトを作る必要がない。

- Perl はインタープリタの形を取るスクリプト言語である。そのため用途に応じて、比較的容易にスクリプトを変更することができる。
- Perl は正規表現をサポートしているツールのうち、現在最も強力で、最も柔軟性を持つと考えられる。
- Perl はインタープリタ型の言語ではあるが、スクリプト全体を解析、コンパイルした後で実行するので、比較的高速な処理を必要とする場合にも十分実用になる。
- Perl はネットワーク・プログラミングにも極めて有用な機能が豊富にある。

ネットワークからの利用も考慮する

Internet が学術系の実験ネットワークの段階を終え、広く一般に門戸を開いて以来、テキスト型データベースも次第に広く公開されるようになってきている。とりわけ World Wide Web (以下 WWW と略す) の爆発的普及により、その傾向はより顕著になってきている。

本プロジェクトでは、著作権者との交渉が完了していないため、当分の間データをネットワーク上で公開することはできない。しかし、データを将来公開することも念頭に入れ、検索用ツールはネットワーク上での操作を意識した設計にすることとした。

現在の段階では、前述の WWW を通じてのデータ検索が最も手軽に行われていると考えられる。そこで本稿でもとりあえず、Perl により記述されたスクリプトを CGI (Common Gateway Interface) として利用した、WWW 上での検索ツールを試作することとする[6]。

特定の Web ブラウザに依存する出力は行わない

WWW による出力は HTML (Hyper Text Mark-up Language) と呼ばれるマークアップ言語により実現される。この HTML も発展過程で特定のブラウザによる独自拡張が乱発されたという経緯がある。今では段階的にではあるものの、統一化の方向に向ってはいるが、なおも残ったままになっている独自拡張もある。

本プロジェクトで試作される Web ページは、こうした特定の Web ブラウザに依存する HTML コードは排除し、現行のすべての Web ブラウザから参照可能なものを目指す。

検索されたデータの出力は検索元ファイル内の TEX, LATEX 形式のままにする

本プロジェクトで作成されたデータは、現行のすべての計算機環境上で利用可能となるよう、極めて汎用性の高い TEX, LATEX での記述方式を利用している。この記述方式は、複数の言語コードが混在する環境、特に日本語などのマルチバイト・コードが存在する環境でも何の問題なく利用でき

[6] Internet や WWW に関して、さまざまな過信や迷信、誤解、技術的問題などがあるのも事実ではあるが、それらを論じることはここでは行わない。しかし、WWW を通してデータを公開する際には、例えば Web Server の運用やネットワーク・セキュリティなどには特に注意を払う必要がある。

る。多国語が混在する環境での言語記述には脆弱な機能しか持たない HTML などと比べ、はるかに優位にあるこの記述方式での出力は、データの汎用性を保持するものとなる。

もちろん検索したデータは、ユーザが利用する個別のシステムに依存したデータへとユーザ自身が変換できるので、TEX や LATEX を利用しなくても問題なくデータの再利用が可能となる。

3.2 基本仕様

先に挙げた概要に基づき、試作する検索用インターフェースの仕様を決めていくことにする。

3.2.1 構成

試作する検索用インターフェースは、基本的に WWW ベースとし、検索等の作業は CGI 及びそこから呼び出される Perl スクリプトにより実現することとする。

まず、全体の構成を以下の部門に区分する。

1. 検索文字列・検索条件等入力ページ (HTML)
2. 検索スクリプト呼出及び出力整形用 CGI (Perl スクリプト)
3. 検索用スクリプト (Perl スクリプト)
4. 検索結果表示ページ (HTML)

次に各部門の基本仕様を記述する。

3.2.2 検索文字列・検索条件等入力ページ (search.html)

データの検索を行う際に、まずこのページから検索に必要なパラメータを入力・選択することになる。当然の事ながら、このページは HTML で記述する。このページから入力・選択できるのは、以下のものとする:

- 検索対象ファイル
- 検索文字列
- 検索の際の大小文字の区別化と同一視化

  「同一視化」を選択項目とし、選択されない場合は自動的に「区別化」が選択されたものとみなす。
- 検索結果の表示モード (行単位/ページ単位)

  「ページ単位」を選択項目とし、選択されない場合は自動的に「行単位」が選択されたものとみなす。

各項目を入力・選択した後、「Search」ボタンを押すことで検索を開始する。

ここで入力・選択させたパラメータは 3.2.3 で述べる CGI スクリプトへ送られる。

なお上記の「検索文字列」の部分には、Perl で利用できる正規表現をそのまま指定し、検索に利用できるものとする。またこのページに検索文字列に関するコメントも記述する。

3.2.3 検索スクリプト呼び出し及び出力整形用 CGI (search.cgi)

CGI として機能する部門で、3.2.2 で指定したパラメータを検索用スクリプトへ送り、帰ってきた検索結果を表示用ページとして整形して Web ブラウザへ返す。Perl で記述され、主として以下のことを行う:

- 3.2.2 で述べた入力用ページで指定したパラメータを受け取る
- 受け取ったパラメータを整形し、3.2.4 で述べる検索スクリプトを呼び出す
- HTML のヘッダその他のタグを出力
- 検索スクリプトからの検索結果を出力

また、CGI としての基本機能は、[11] 所収の Perl スクリプト cgiparse.pl を一部改良し、利用することで実現する。

3.2.4 検索用スクリプト (tsearch.pl)

受け渡されたパラメータをもとに、指定されたテキスト型ファイルから文字列検索を行う。Perl のスクリプトで記述され、以下の内容を標準出力へ出力する:

- 行単位出力モードの場合
  - 検索ファイル名
  - ページ番号
  - (ヒットした文字列が註の中にある場合は) 註番号
  - ヒットした文字列のある行の行番号
  - 当該ヒット文字列を含む行 (当該ヒット文字列および行番号は、太字または反転文字で強調)、同一ページ内で複数行にまたがる場合も当該行をすべて表示
- ページ単位出力モードの場合
  - 検索ファイル名
  - ページ番号
  - テキスト行番号
  - 当該ページテキスト (当該ヒット文字列および行番号は、太字または反転文字で強調)

なお、太字または反転文字で強調表示するための制御文字列は、HTML のタグばかりでなく、ANSI エスケープ・シーケンスも出力できるようにしておく (パラメータによりどちらかを選択・指示する)。こうした設計にした理由は、このスクリプトが完全に Web に依存したものではなく、独立した Perl スクリプトとしても利用できるからである。CGI が利用できない環境下にある場合でも、この Perl スクリプトを利用することで、検索作業ができるのである。

3.2.5 検索結果表示ページ

3.2.4 での検索スクリプトにより得られた結果を 3.2.3 で述べた CGI の制御に基づき表示する。Web ブラウザが GUI ベースのものである場合、表示される検索結果はコピー & ペーストの機能で別ファイルへ張り付けることもできる。もちろん Web ブラウザのファイル保存機能を利用することでも、検索結果のファイルへの保存は可能となる[7]。

4 出来上がった Web ページと表示

3.2 での基本仕様にもとづき、システムを作成・動作試験を行った。なおここで使用したソフトウェア環境の概要は次の通りである[8]:

- OS: Linux (Kernel Ver. 2.0.31 + Slackware 3.3 + PJE-0.1beta)
- HTTP Server: Apache Ver. 1.2.0
- Perl: Perl Ver. 5.004 + Japanization Patch 4
- Window System: X Window System (XFree86 3.3.1 + fvwm + afterstep)
- Web ブラウザ: Netscape Ver. 3.01 (for Linux)

次に作成された Web ページやその出力結果の実例を示すことにする。

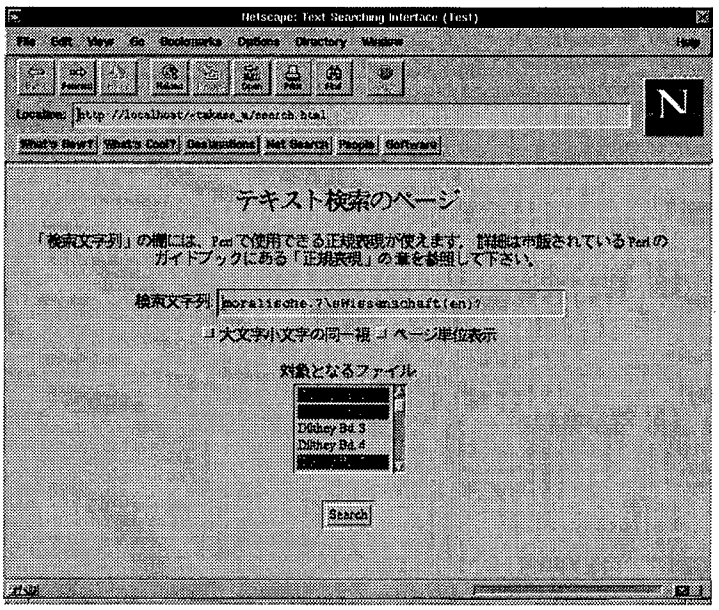
[7] ただしその場合、HTML のタグが若干混入することになるが、データ再利用の妨げになる程のものではない、と考えられる。

[8] 当然の事ながら、ここで試作・試験したものは、システムに依存したものではないので、他の環境下でも必要なソフトウェアをそろえておけば、同様の結果を得ることができるのは言うまでもない。特に Web ブラウザには、ここでは Netscape を使用したが、Lynx のような文字ベースのブラウザでもほぼ問題なく使用できることが確認されている。

4.1 基本使用例

4.1.1 検索文字列・検索条件等入力ページ

ユーザはまずこのページから検索条件等を入力する。



この例では、各項目は次のように入力・選択されている:

検索文字列: "`moralische.?\sWissenschaft(en)?`"

これは、"moralische Wissenschaft(en)" という語句の各変化形を検索することを意味する[9]。"\s" は [`\n\r\f\t`] の省略記法である。これにより、同一レコード内の複数行にまたがる場合も検索が可能となる。

大小文字: 区別して検索

出力モード: 行単位

対象ファイル: Dilthey 著作集第 1 巻、第 2 巻及び第 5 巻

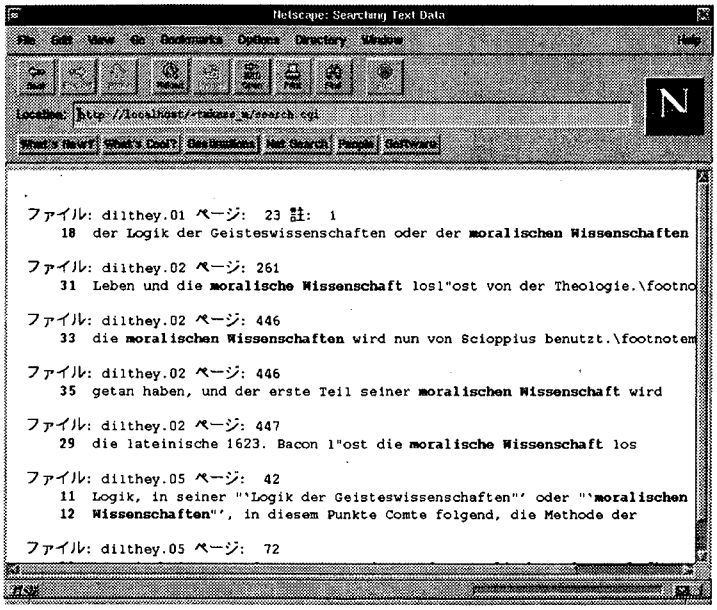
4.1.2 検索結果表示ページ

4.1.1 で入力・選択された条件をもとにデータ・ファイルに対して検索作業が行なわれる。その結果の表示状態は「行単位出力モード」と「ページ単位出力モード」とでは異なるが、ここでは、それぞれの出

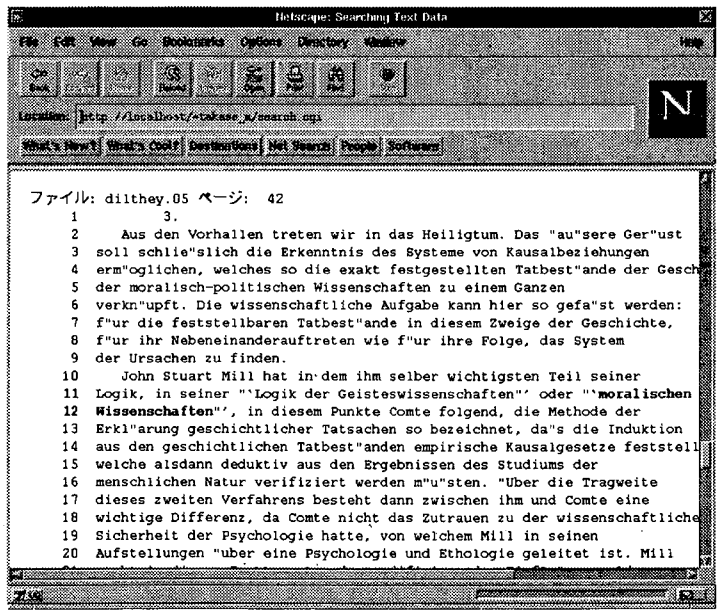
[9] 本来はもう少し精密に記述する方が望ましいが、ここではあえて簡略な記述にとどめておく。

力結果の実例を示す。

行単位出力モード



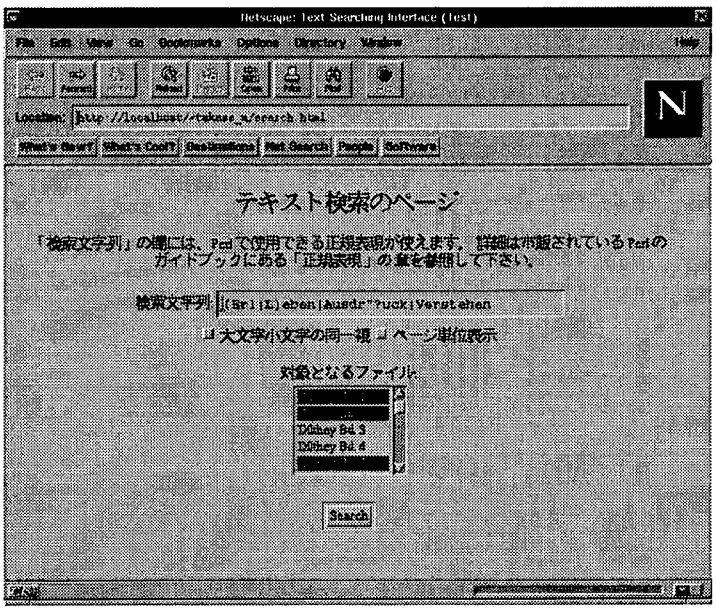
ページ単位出力モード



4.2 応用例

次に複数種類の語をまとめて検索する例を示す。

4.2.1 検索文字列・検索条件等入力ページ



この例では、各項目は次のように入力・選択されている:

検索文字列: **"(Erl|L)eben|Ausdr"?uck|Verstehen"**

この検索文字列により、„Erleben", „Leben", „Ausdruck", „Verstehen" という各語のいずれかがヒットする (変化形も含む)。

大小文字: 区別して検索

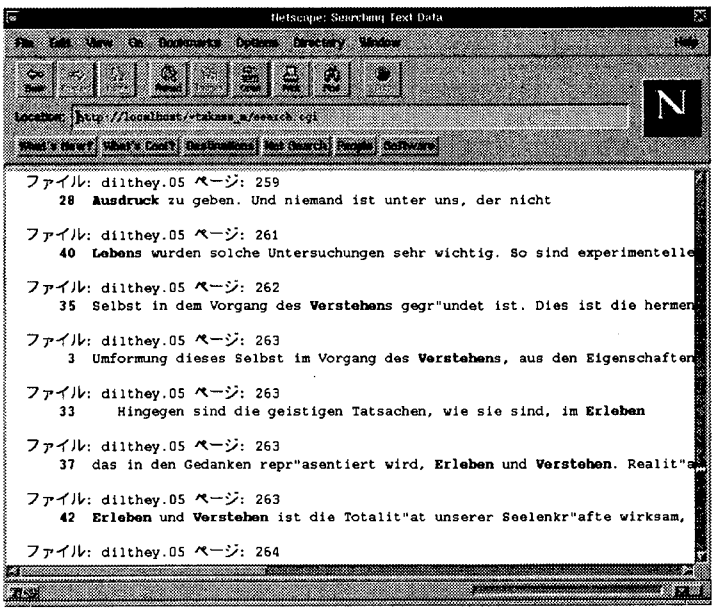
出力モード: 行単位

対象ファイル: Dilthey 著作集第 1 巻、第 2 巻及び第 5 巻

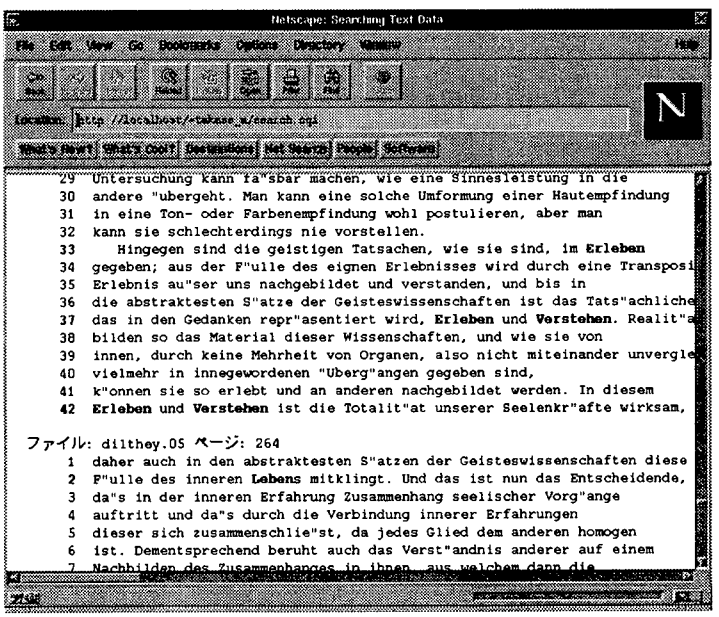
4.2.2 検索結果表示ページ

4.1.2 と同様、4.2.1 で指定した条件により行なわれた検索の結果は、その指定により「行単位出力モード」と「ページ単位出力モード」の二通りの表示方式がある。ここでもその二通りの実例を示す。

行単位出力モード



ページ単位出力モード



5 結果と今後の課題

今回、テキスト型データ・ベースからの検索用インターフェースを試作、動作試験を行った結果、次のようなことが確認できた。

- Perl の持つ正規表現の機能は予想した通り、テキスト検索には極めて効果的である。
  正規表現による検索文字列の記述は、誰にでもできるレベルから、かなりの熟練を要するレベルまであるものの、比較的簡単な記述でもある程度高い確率でヒットするものが多く、十分な効果が期待できる。
- 仮に一度で希望通りにヒットしなくとも、ここで試作されたようなインターフェースを通して何度か繰り返し検索を行うことで、十分に効果的な検索が行えるようになる。
- サーバ・マシンとは LAN で結ばれているクライアント・マシンからも試作した検索システムを利用してみたところ、サーバ・マシン上でクライアントを操作するのとほとんど変りなく結果を得ることができるのを確認した。特に動作速度は複数のファイルを検索対象にしても、出力には時間はさほどかからなかった[10]。

また、今後の課題として次のような点が挙げられる:

- 検索用ページの整備
- 作成したスクリプト類の改良
- データ公開に向けた著作権者との交渉
- 作成したテキスト型データのエラー除去と整備

今後もこのような課題を解決しながら、よりよい検索システムを目指す必要があろう。

[10] もちろんこれは外部へ接続していない LAN 環境下で実験を行なったせいもあるが、CGI や検索スクリプトが現段階での最小コストで動作していることも一因となっていると考えられる。いずれにしてもネットワークを通してのデータ検索には、それなりに高速で、ある程度の安全性の確保されたネットワーク環境が必要なことは、言うまでもない。

参考文献

[1] 磯野　康孝/蔵守　伸一 共著:「HTML ハンドブック」, ナツメ社, 1996.

[2] 轡田　収/高瀬　誠:「OCR によるデータベース作成」, 学習院大学計算機センター年報 Vol. 16, 1995 所収.

[3] 小山　裕司/斎藤　靖/佐々木　宏/中込　知之 著:「UNIX 入門 — フリーソフトウェアによる最新 UNIX 環境」, 株式会社 トッパン, 1996.

[4] 小山　裕司/斎藤　靖/佐々木　宏/中込　知之 著:「Linux 入門」, 株式会社 トッパン, 1996.

[5] 坂本　文 著:「たのしい UNIX — UNIX への招待 —」, アスキー出版局, 1991.

[6] 坂本　文 著:「続・たのしい UNIX — シェルへの招待 —」, アスキー出版局, 1993.

[7] 志村　拓/鷲北　賢/西村　克信 共著:「AWK を 256 倍使うための本」, アスキー出版局, 1993.

[8] 高瀬　誠:「計算機による欧文テキスト処理について— 機種依存しない統一的処理についての一考察 —」, 学習院大学大学院ドイツ文学語学研究 第 18 号, 1994 所収.

[9] 高瀬　誠:「Text-Korpus からの検索について— 現代ドイツ語動詞の検索用正規表現の試み —」, 学習院大学ドイツ文学会 研究論集 1, 1997 所収.

[10] 中野　賢 著:「日本語 LaTeX 2ε ブック」, アスキー出版局, 1996.

[11] 西田　知博/川本　芳久 共著:「CGI 入門講座 WWW サーバーを料理する!」, オーム社, 1997.

[12] 野寺　隆志　著:「楽々LaTeX・第 2 版」, 共立出版, 1994.

[13] 藤田　眞作　著:「LaTeX 2ε 階梯」, アジソン・ウェスレイ・パブリッシャーズ・ジャパン, 1996.

[14] 前田　薫/小山　裕司/斉藤　靖/布施　有人 共著:「新 Perl の国へようこそ」, サイエンス社, 1996.

[15] やまだ　あきら/はね　ひでや　共著:「Networking Linux」, アスキー出版局, 1997.

[16] Aho, A. V./Kernighan, B. W./Weinberger, P.J. 共著/足立　高徳 訳:「プログラミング言語 AWK」, 株式会社 トッパン, 1989.

[17] Bryan, Martin 著/山崎　俊一 監訳/福島　誠 訳:「SGML 入門」, アスキー出版局, 1991.

[18] Friedl, Jeffery E. F.: Mastering Regular Expressions, O'Reilly & Associates, Inc., 1997.

[19] Gundavaram, Shishir 著/田辺 茂也 監訳/株式会社エディックス 訳:「CGI プログラミング」, オライリー・ジャパン, 1996.

[20] Kirch, Olaf 著/福島 隆一・高尾 哲康 共訳/山崎 康宏 監訳:「Linux ネットワーク管理」, オライリー・ジャパン, 1996.

[21] Knuth, D. E. 著/鷺谷 好輝 訳/斉藤 信男 監修:「TEX ブック」, アスキー出版局, 1989.

[22] Knuth, Donald Ervin: The TEXbook (COMPUTERS & TYPESETTING Volume A), Addison Wesley Publishing Company, 1986.

[23] Lamport, Leslie: LATEX: A Document Preparation System User's Guide & Reference Manual, Addison Wesley Publishing Company, 1986.

[24] Laurie, Ben/ Peter Laurie 共著/田辺 茂也 監訳/三代川 信義 訳:「Apache ハンドブック」, オライリー・ジャパン, 1997.

[25] Lunde, Ken 著/春遍 雀来・鈴木 武生 訳:「日本語情報処理」, O'Reilly and Associates, Inc., ソフトバンク株式会社, 1995.

[26] Raichle, Bernd: „Kurzbeschreibung — german.sty (Version 2.5) 1. Mai 1997 (für german.sty Version 2.5d)".

[27] Schwarz, Randal L./Tom Christiansen: Lerning Perl, 2nd Edition, O'Reilly and Associates, Inc., 1997[2].

[28] Wall, Larry/Tom Christiansen/Randal L. Schwarz 著/近藤 嘉雪 訳:「プログラミング Perl 改訂版」, オライリー・ジャパン, 1997.

付録. スクリプト等ソースリスト

以下に挙げるソースーリストは、本プロジェクトにおいて試作した Web ページと Perl によるスクリプトである。それぞれの仕様については、3.2 を参照されたい[11]。

A. 検索文字列・検索条件等入力ページ (search.html)

```
1  <!-- テキスト型データ検索用ページ (search.html Ver.1.00) -->
2  <!-- Copyright (C) TAKASE Makoto, 1997. -->
3  <HTML>
4  <HEAD>
5  <TITLE>Text Searching Interface (Test) </TITLE>
6  </HEAD>
7  <BODY>
8  <CENTER>
9  <H1> テキスト検索のページ </H1>
10
11 <FONT SIZE=3>
12          「検索文字列」の欄には、Perl で使用できる正規表現が使えます。
13          詳細は市販されている Perl のガイドブックにある「正規表現」の
14          章を参照して下さい。
15 </FONT>
16
17 <FORM METHOD=POST ACTION="http://localhost/~takase_m/search.cgi">
18 <FONT SIZE=3> 検索文字列:</FONT> <INPUT TYPE=TEXT SIZE=40 NAME="searchstr"><BR>
19
20 <INPUT TYPE=CHECKBOX NAME=ingore_upcase><FONT SIZE=3> 大文字小文字の同一視 </FONT>
21
22 <INPUT TYPE=CHECKBOX NAME=pagemode><FONT SIZE=3> ページ単位表示 </FONT>
23
24 <P>
25 <FONT SIZE=3>
26     対象となるファイル:
27 </FONT><BR>
```

[11] 各スクリプトの著作権はそれぞれのソースの先頭に示した通り、高瀬　誠 (Makoto.Takase@gakushuin.ac.jp) に有るものとする。各スクリプトは再利用・改変等を自由に行なう事ができるが、それにより生じたいかなる損害に対しても原著作者は責任を負わない。改変に際しては、スクリプト内の著作者表示の部分は消去せずに改変者の名前及び変更日付を追加していく事。
なお、各スクリプトのバグなどについては、原著作者へ連絡されたい。

```
28 <SELECT NAME=files SIZE=5 MULTIPLE>
29     <OPTION VALUE="dilthey.*">   Dilthey Alle Bde.
30     <OPTION VALUE="dilthey.01">   Dilthey Bd. 1
31     <OPTION VALUE="dilthey.02">   Dilthey Bd. 2
32     <OPTION VALUE="dilthey.03">   Dilthey Bd. 3
33     <OPTION VALUE="dilthey.04">   Dilthey Bd. 4
34     <OPTION VALUE="dilthey.05">   Dilthey Bd. 5
35     <OPTION VALUE="dilthey.06">   Dilthey Bd. 6
36     <OPTION VALUE="dilthey.07">   Dilthey Bd. 7
37     <OPTION VALUE="dilthey.08">   Dilthey Bd. 8
38     <OPTION VALUE="dilthey.09">   Dilthey Bd. 9
39     <OPTION VALUE="dilthey.10">  Dilthey Bd. 10
40     <OPTION VALUE="dilthey.11">  Dilthey Bd. 11
41     <OPTION VALUE="dilthey.12">  Dilthey Bd. 12
42     <OPTION VALUE="dilthey.13a"> Dilthey Bd. 13/1
43     <OPTION VALUE="dilthey.13b"> Dilthey Bd. 13/2
44     <OPTION VALUE="dilthey.14a"> Dilthey Bd. 14/1
45     <OPTION VALUE="dilthey.14b"> Dilthey Bd. 14/2
46     <OPTION VALUE="dilthey.15">  Dilthey Bd. 15
47     <OPTION VALUE="dilthey.16">  Dilthey Bd. 16
48     <OPTION VALUE="dilthey.17">  Dilthey Bd. 17
49     <OPTION VALUE="dilthey.18">  Dilthey Bd. 18
50     <OPTION VALUE="dilthey.19">  Dilthey Bd. 19
51     <OPTION VALUE="dilthey.20">  Dilthey Bd. 20
52     <OPTION VALUE="dilthey.21">  Dilthey Bd. 21
53 </SELECT><P>
54 <INPUT TYPE=SUBMIT VALUE="Search">
55
56 </FORM>
57 </CENTER>
58
59 </BODY>
60 </HTML>
```

B. 検索スクリプト呼び出し及び出力整形用 CGI (search.cgi)

```
#!/usr/bin/perl
#
# CGI Script for Text Searching Ver.1.00
# Copyright (C) TAKASE Makoto, 1997.
#
# cgiparse.pl は、西田　知博/川本　芳久 共著:
# 「CGI 入門講座 WWW サーバーを料理する!」,
# オーム社, 1997. から一部改良して利用。
#
require 'cgiparse.pl';

BEGIN{
    $IgnoreUpcaseOption = '';
    $PageModeOption = '';
    $HTMLOption = '-h';
    $SearchCommand ='./tsearch.pl';
}
&cgi'Parse();

$searchstr= $cgi'name{'searchstr'};
$ingore_upcase = $cgi'name{'ingore_upcase'};
if ($ingore_upcase ne ''){
        $IgnoreUpcaseOption = '-i';
}
$pagemode= $cgi'name{'pagemode'};
if ($pagemode ne ''){
        $PageModeOption = '-p';
}
@files = &cgi'Value('files');
foreach $i (@files){
   $Targetfiles .= sprintf(" %s", $i);
}

print &cgi'HTTPHeader();

```

```
36 print &cgi'SimpleTitle('Searching Text Data');
37 print "<BODY bgcolor=#ffffff>\n";
38
39 $CommandLine = sprintf("%s %s %s %s '%s' %s",
40                 $SearchCommand , $IgnoreUpcaseOption ,
41                 $PageModeOption , $HTMLOption , $searchstr , $Targetfiles);
42 print "<pre>\n";
43 open(OUT, "$CommandLine |");
44 while (<OUT>){
45         print $_;
46 }
47 close(OUT);
48 print "</pre>\n";
49
50 print &cgi'SimpleEnd();
```

C. 検索用スクリプト (tsearch.pl)

```
 1 #!/usr/bin/jperl
 2 #
 3 #               Text Search Script Ver. 1.0
 4 #               Copyright (C) TAKASE Makoto, 1997.
 5 #
 6 # 複行レコードからのデータ検索
 7 # Usage:
 8 #        tsearch.pl [-i] [-p] [-g|-e] [-h] 'RegExp' filename
 9 #
10 BEGIN{                          ### 初期設定 ###
11         $/="";                                  # 複行レコード
12                                                 # デフォルト値設定
13         ($IgnoreUpcaseMode, $RegexpMode , $PageMode)=(0, 0, 0);
14         ($PraeCode, $PostCode)=("\e[7m", "\e[m");
15         ($CheckPraeCode, $CheckPostCode)=("\e\\[7m", "\e\\[m");
16         ($FileNameMark, $PageMark, $FootNoteString)=
17                                 ('ファイル:', 'ページ:', '註:');
18         $FootNoteText='footnotetext';
```

```
19          while (@ARGV[0] =~ /^\-/){              # パラメータ取得
20                  $Param=shift(@ARGV);
21                  if ($Param eq "-i"){    # Ignore Upcase Mode
22                          $IgnoreUpcaseMode=1;
23                  }elsif($Param eq "-p"){ # Page Output Mode
24                          $PageMode=1;
25                  }elsif($Param eq "-e"){ # English Language
26                          ($FileNameMark, $PageMark, $FootNoteString)=
27                                  ('File:', 'Page:', 'Note:');
28                  }elsif($Param eq "-g"){ # German Language
29                          ($FileNameMark, $PageMark, $FootNoteString)=
30                                  ('Datei:', 'Seite:', 'Anm.:');
31                  }elsif($Param eq "-h"){ # HTML Output
32                          ($PraeCode, $PostCode)=('<b>', '</b>');
33                          ($CheckPraeCode, $CheckPostCode)=('<b>', '</b>');
34                  } else {die "Parameter Error!"}
35          }
36          ($Regexp, @Files)=@ARGV;
37          @ARGV=@Files;
38  }
39
40  while (<>){                 ### main ###
41          $Filename=$ARGV;                        # ファイル名取得
42          $Data=$_;
43                                  # 検索 & 出力
44           if (&Search){
45                  @PageText=split("\n", $Data);  # リストへ
46                  $PageNo=$PageText[0];          # ページ数取得
47                  $Lines=@PageText-1;            # 当該ページ行数取得
48                  if ($PageMode){
49                          &PrintPageMode;
50                  } else {
51                          &PrintLineMode;
52                  }
53          }
54  }
55
```

```
56  sub Search{                    ### 文字列検索ルーチン ###
57          if ($IgnoreUpcaseMode){
58                  $ReturnCode=($Data =~ s/($Regexp)/$PraeCode$1$PostCode/ig);
59          }else{
60                  $ReturnCode=($Data =~ s/($Regexp)/$PraeCode$1$PostCode/g);
61          }
62          if ($ReturnCode > 0){$ReturnCode=1};
63          $ReturnCode;
64  }
65
66  sub CheckFootNote{          ### 註のチェック ###
67          if ($PageText[$TempLineNo] =~ /$FootNoteText/){
68                  $LineNo=0;
69                  $PageText[$TempLineNo] =~ /\[([\w\*]*)\]/;
70                  $FootNoteNo = $1;
71                  $FootNoteExp=sprintf("%s %2s", $FootNoteString, $FootNoteNo);
72          }elsif ($PageText[$TempLineNo] =~ /^}$/){
73                  $LineNo=0;
74          }else {
75                  $LineNo++;
76          }
77  }
78
```

```
sub PrintPageMode{          ### ページ単位出力モードルーチン ###
        $LineNo=0; $FootNoteExp=''; $Hit=0;
        printf("%s %s %s %3s\n", $FileNameMark, $Filename, $PageMark, $PageNo);
        for ($TempLineNo=1; $TempLineNo <= $Lines; $TempLineNo++){
                &CheckFootNote;
                $PrintLine=$PageText[$TempLineNo];
                if ($PageText[$TempLineNo] =~ /$CheckPraeCode/){$Hit=1;}
                if ($Hit >= 1){
                        if ($Hit == 1){
                                $LineNoExp=sprintf("%s%6s%s  ",
                                        $PraeCode, $LineNo, $PostCode);
                        } else{
                                $LineNoExp=sprintf("%s%6s%s  %s",
                                        $PraeCode, $LineNo, $PostCode, $PraeCode);
                        }
                        if ($PageText[$TempLineNo] !~ /$CheckPostCode/){
                                $PrintLine .= $PostCode;
                        }
                        $Hit++;
                } elsif ($LineNo == 0){
                        $LineNoExp='';
                } else{
                        $LineNoExp=sprintf("%6s  ", $LineNo);
                }
                printf("%s%s\n", $LineNoExp, $PrintLine);
                if ($PageText[$TempLineNo] =~ /$CheckPostCode/){$Hit=0;}
        }
        print "\n";
}
```

```
109 sub PrintLineMode{          ### 行単位出力モードルーチン ###
110         $LineNo=0; $FootNoteExp=''; $Hit=0;
111         for ($TempLineNo=1; $TempLineNo <= $Lines; $TempLineNo++){
112                 &CheckFootNote;
113                 $PrintLine=$PageText[$TempLineNo];
114                 if ($PageText[$TempLineNo] =~ /$CheckPraeCode/){$Hit=1;}
115                 if ($Hit >= 1){
116                         if ($Hit == 1){
117                                 $LineNoExp=sprintf("%s%6s%s  ",
118                                         $PraeCode, $LineNo, $PostCode);
119                                 printf("\n%s %s %s %3s %s\n",
120                                         $FileNameMark, $Filename,
121                                         $PageMark, $PageNo, $FootNoteExp);
122                         } else{
123                                 $LineNoExp=sprintf("%s%6s%s  %s",
124                                         $PraeCode, $LineNo, $PostCode, $PraeCode);
125                         }
126                         if ($PageText[$TempLineNo] !~ /$CheckPostCode/){
127                                 $PrintLine .= $PostCode;
128                         }
129                         $Hit++;
130                         printf("%s%s\n", $LineNoExp, $PrintLine);
131                 } elsif ($LineNo == 0){
132                         $LineNoExp='';
133                 }
134                 if ($PageText[$TempLineNo] =~ /$CheckPostCode/){$Hit=0;}
135         }
136 }
```