研究報告

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(2) — 他形式のデータに対する応用 —

轡田 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い、作成されたデータから必要なデータを検索する方法を考察してきた[1]。本稿では、ここで一度 Dilthey 著作集を離れ、同様の記述方式でありながら構造が若干違うデータを扱い、そこから必要なデータを検索する点について考察してみる。

2 データの基本仕様

2.1 対象となるデータ

これまでは本プロジェクトでは、Wilhelm Dilthey 著作集をデータとしてきた。このデータでは原テキストの構成から、[1], [2] で述べた通り、原テキストの 1 ページを 1 レコードの単位として取扱ってきた。同様の形式を取るものであれば、他のデータでも先の [2] で示したスクリプト等を利用して、同様の取扱いが可能となる。

しかし原テキストの構成上、ページ単位ではまとめられないテキストも存在する。例えば中高ドイツ語時代に作成された数々の叙事詩がこれにあたる。こうしたテキストのデータの中には、ページ単位で取扱う事はできず、これまで用意したスクリプトをそのままの形で利用すると、データ全体が 1 ページと解釈されてしまうものもある。その場合、検索用サーバのメモリを徒に使用するばかりで、効率的な検索は難しくなってくる。

そこで本稿では、その様なデータも試験的に用意し、そこから必要な部分を検索・抽出するための Perl スクリプトを試作する事とする。試験的に用意されるデータは、Hartmannn von Aue 作の „Der arme Heinrich“ の本文全てとする[2]。

† 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] [10] に基づいてデータは作成され、データ全体の行数はおよそ 1550 行弱となる。

2.2 データの記述方式と構造

検索元になるテキスト型データの記述方式と構造は、若干の点を除いて基本的な点では [1] とあまり変わらない。ここではその概略を確認するに留めておく。

- オリジナルテキストの 1 行がデータの 1 行に相当する
- 一行のデータは **<tab code><text>** という形式にする
- データには 5 行毎にオリジナルテキストの行番号を付加する
- その際の一行のデータ形式は **<行番号><tab code><text>** とする
- # を行頭に付加した場合はその行はコメント行とする

なお、データ内の特殊文字の記述方式は、これまでと同様に TEX 並びに LATEX のドイツ語用スタイルファイル german.sty での記述方式を基準とし、そこで定義されているもの以外についても TEX 及び LATEX での標準的記述方式に従う事とする。

これにより、当該のデータはおおよそ次の様なものとなる:

```
        Ein ritter s\^o gel\^eret was
        daz er an den buochen las
        swaz er dar an geschriben vant:
        der was Hartman genant,
5       dienstman was er zOuwe.
        er nam im manige schouwe
        an misl\^{\i}chen buochen:
        dar an begunde er suochen
        ob er iht des vunde
10      d\^a mite er sw{\ae}re stunde
        m"ohte senfter machen,
        und von s\^o gewanten sachen
                                    以下省略
```

3 試作スクリプトの基本仕様

試作されるスクリプト (ここではとりあえずその名前を atsearch.pl としておく) の基本的な仕様は次の通りとする。なお、このスクリプトは試験的なもののため、原段階ではコマンドラインからの起動のみを意図しているが[3]、将来的に CGI からの起動へも対応する事を視野に入れて設計しておく:

- 起動オプション、検索文字列、検索対象ファイルはコマンドラインから指定
- オプションには以下のものを指定する事ができる
  - -i: 大文字・小文字の区別の無効化
  - -h: HTML による太字強調 (... を利用)
  - -n: 強調出力の停止
  - -数字: ヒット文字列を含む行の前後 n 行も出力
- 検索文字列には '...' の様に ' で括られ、Perl で利用できる正規表現が全て利用できる様にする
- そのため複数行にわたる検索にも対応できるようにしておく
- 出力形式は GNU grep のそれを手本としながら、それを補う形とし、以下の通りとする:
  - 出力形式は<ファイル名><ヒットマーク><行数番号><当該行テキスト> とする
  - ヒットマークは検索文字列を含む行では : に、そうでない行は - とする
  - 検索文字列にヒットした部分とその行数番号には、太字もしくは反転などで強調して出力する事を基本とする。その表示方法は以下の通り:
    - オプションなし (デフォルト) では ANSI エスケープシーケンスによる反転強調
    - -h オプションで HTML による太字強調
    - -n オプションで強調出力の停止
  - 出力する各行に対して行数番号を加える事で Text 内の位置を明確にする

ソースコードは付録に掲載しておく事とする。

[3] 今回は Linux Slackware 3.4 + PJE-0.1 の上での動作の様子を示す事とする。しかし周知の通り、Perl スクリプトはその動作に対して、特に計算機環境を限定するものではないので、他の計算機環境でもここで挙げるのと同様な結果を得る事ができよう。

4 動作結果

試作したスクリプト atsearch.pl と、対象テキストである „Der arme Heinrich“ (ファイル名 heinr.txt) を利用して、幾つかの検索とその出力の実例をここで挙げる事にする。

4.1 語レベルでの検索

まず、語形変化をしている語を正規表現で検索する事にする。ここでは helfen という語とその各変化形を当該テキスト内から検索する。その際、検索文字列を含む行のみを出力すると、次の様な結果となる:

```
/home/takase_m>./atsearch.pl -n 'h([eiao]|"?u)lf[est]*n*' heinr.txt
heinr.txt:   222 d\^a von muoz iu diu helfe m\^{\i}n
--
heinr.txt:   333 ouch half in s\^ere daz diu kint
--
heinr.txt:   919 daz man iu gehelfen mac,
--
heinr.txt:   935 m"ohtest\^u, d\^u h"ulfest mir.
--
heinr.txt:  1109 ir h"ulfe des tages der t\^ot
--
heinr.txt:  1317 s\^o helfe mir got, si h\^ant gelogen.
--
heinr.txt:  1520 des helfe uns got. \^amen.
--
```

これはもちろん、GNU grep と同じく当該文字列を含む行とその前後 3 行を出力させる事も可能である:

```
/home/takase_m>./atsearch.pl -n -3 'h([eiao]|"?u)lf[est]*n*' heinr.txt
heinr.txt-   219 mit deheinem liste erwerben,
heinr.txt-   220 ich enlieze iuch niht verderben.
heinr.txt-   221 nu enmac des leider niht s\^{\i}n:
heinr.txt:   222 d\^a von muoz iu diu helfe m\^{\i}n
heinr.txt-   223 durch alle n\^ot s\^{\i}n versaget.
heinr.txt-   224 ir m"ueset haben eine maget
heinr.txt-   225 diu vollen manb{\ae}re
--

(以下省略)
```

4.2 複数行にまたがる検索

Perl の正規表現を利用した検索が持つ大きな特徴の一つに、複数行にまたがる検索が可能であるという点が挙げられる。ここで試作した atsearch.pl もそれに対応している。ここではその一例として、(al)sô ··· daz... という語句を検索してみる:

```
/home/takase_m/>./atsearch.pl -n '(al)?s\\\^o\b.*\ndaz' heinr.txt
heinr.txt:     1 Ein ritter s\^o gel\^eret was
heinr.txt:     2 daz er an den buochen las
--
heinr.txt:    12 und von s\^o gewanten sachen
heinr.txt:    13 daz gotes \^eren t"ohte
--
heinr.txt:   212 ich mache iuch mir als\^o holt
heinr.txt:   213 daz ir mich harte gerne ernert.'
--
heinr.txt:   216 'und w{\ae}re der arzen\^{\i}e als\^o
heinr.txt:   217 daz man si veile vunde

以下省略
```

当然の事ながら、これにも前後の行を付加して出力する事もできる:

```
./atsearch.pl -n -2 '(al)?s\\\^o\b.*\ndaz' heinr.txt
heinr.txt:     1 Ein ritter s\^o gel\^eret was
heinr.txt:     2 daz er an den buochen las
heinr.txt-     3 swaz er dar an geschriben vant:
heinr.txt-     4 der was Hartman genant,
--
heinr.txt-    10 d\^a mite er sw{\ae}re stunde
heinr.txt-    11 m"ohte senfter machen,
heinr.txt:    12 und von s\^o gewanten sachen
heinr.txt:    13 daz gotes \^eren t"ohte
heinr.txt-    14 und d\^a mite er sich m"ohte
heinr.txt-    15 gelieben den liuten.
--
heinr.txt-   210 und dar zuo versprechen
heinr.txt-   211 beidiu m\^{\i}n silber und m\^{\i}n golt,
heinr.txt:   212 ich mache iuch mir als\^o holt
heinr.txt:   213 daz ir mich harte gerne ernert.'
heinr.txt-   214 'mir w{\ae}re der wille unerwert'
heinr.txt-   215 sprach der meister aber d\^o:
--

以下省略
```

5　最後に

以上見てきた通り、ここで試作されたスクリプトを利用する事により、ページ単位では区切れないテキスト型データについても、かなり柔軟な検索ができる事が明らかになった。ここでは例示はしなかったが、このスクリプトでは、当該検索文字列部分とその行の行番号が強調表示される機能も付いている。今後はCGIから利用する事もある程度視野に入れていく必要もあろう。

今後もここで試作したスクリプトを改良していくと共に、様々なデータを用意し、柔軟なかつ多様な検索・調査を可能にしていくことを目指していきたい。

参考文献

[1] 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.

[2] 轡田 収／高瀬 誠: 「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」, 学習院大学計算機センター年報 Vol. 18, 1997 所収.

[3] 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.

[4] 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.

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

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

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

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

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

[10] Hartmann von Aue: Der arme Heinrich, Herausgegeben von Hermann Paul, 15., neubearbeitete Auflage, besorgt von Gesa Bonath, Altdeutsche Textbibliothek Nr. 3, Max Niemeyer Verlag, Tübingen, 1984.

[11] 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.

[12] Lamport, Leslie 著/ Edgar Cooke・倉沢 良一 監訳/大野 俊治・小暮 博道・藤浦 はる美 訳: 「文書処理システム LaTeX」, アスキー出版局, 1990.

[13] Raichle, Bernd: „Kurzbeschreibung german.sty und ngerman.sty (Version 2.5) 8. Juli 1998 (für die Version 2.5e)".

[14] Schwarz, Randal L./Tom Christiansen 著/近藤 嘉雪 訳: 「初めての Perl 第 2 版」, オライリー・ジャパン, 1998.

[15] 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. 試作スクリプトソースリスト

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 Perl によるスクリプトである。仕様については、3 を参照されたい[4]。

```
#!/usr/bin/jperl
#
#       Yet another Text Search Script Ver. 0.1
#               Copyright (C) TAKASE Makoto, 1997.
#
# useage: ./atsearch.pl <Options> '<Regexp>' <Files..>
#
BEGIN{                                          ### 初期設定 ###
        ($PraeCode, $PostCode)          = ("\e\[7m", "\e\[m");
        ($CheckPraeCode, $CheckPostCode) = ("\a", "\e");
        ($True, $False)   = (1, 0);
        $IgnoreUpcaseMode = $False;
        $BufferSize       = 50;
        $HalfBufferSize   = $BufferSize/2;
        $OutBufferSize    = 0;
        $OutputLine       = "";
        $Ausgabe          = $False;
        $Hit              = $False;
        $PraeOn           = $True;
        $PostOn           = $False;
        $#PraeOutBuffer   = -1;
        $#PostOutBuffer   = -1;
        $HitLast          = 0;
        $WorkedLine       = 0;
        while (@ARGV[0] =~ /^\-/){        ### パラメータ取得 ###
                $Param=shift(@ARGV);
                if ($Param eq "-i"){                # Ignore Upcase Mode
                        $IgnoreUpcaseMode=$True;
                } elsif ($Param eq "-h"){        # HTML Output
                        ($PraeCode, $PostCode)=('<b>', '</b>');
                } elsif ($Param eq "-n"){        # Normal Output
                        ($PraeCode, $PostCode)=('', '');
                } elsif ($Param =~ /\-([0-9].*)/){
                        $OutBufferSize = $1;        # 前後 n 行出力
                } else {die "Parameter Error!"}
        }
        ($Regexp, @Files)= @ARGV;
        @ARGV = @Files;
}
```

[4] このスクリプトは、今なお試作段階のものであり、完全な動作テストを行っていない。スクリプトの著作権はソースの先頭に示した通り、原著作者である高瀬　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、このスクリプト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。

```
40                                      ### main ###
41  foreach $FileName (@Files){
42          $Text=""; $i=0; $#LineNums=0; $LineNo=0;
43          open(IN, $FileName);
44          while(<IN>){
45                  unless (/^#/){
46                          if ($i++ < $BufferSize-1){
47                                  &AddData;
48                                  if (eof(IN)){
49                                          if (&Search){&OutputData;}
50                                  }
51                          } else {
52                                  &AddData;
53                                  if (&Search){&OutputData;}
54                                  @DataX = split(/\n/, $Text);
55                                  @DataX = splice(@DataX, $HalfBufferSize);
56                                  $Text  = sprintf("%s\n", join("\n", @DataX));
57                                  @LineNums = splice(@LineNums, $HalfBufferSize);
58                                  $i = $HalfBufferSize;
59                          }
60                  }
61          }
62          close(IN);
63  }
64
65  sub Search{                             ### 文字列検索ルーチン ###
66          if ($IgnoreUpcaseMode){
67                  $ReturnCode=($Text =~ s/($Regexp)/$CheckPraeCode$1$CheckPostCode/ig);
68          }else{
69                  $ReturnCode=($Text =~ s/($Regexp)/$CheckPraeCode$1$CheckPostCode/g);
70          }
71          if ($ReturnCode > 0){$ReturnCode=1};
72          $ReturnCode;
73  }
74
75  sub AddData{                            ### データ読込ルーチン ###
76          my($TmpText, $LineX, $TmpLineNo);
77          $LineX=$LineNo;
78          ($TmpLineNo, $TmpText) = /(\w*)\t([\w\W]*)/;
79          if ($TmpLineNo eq ""){
80                  $LineNo = $LineX + 1;
81          }elsif ($TmpLineNo =~ /[a-z]/){
82                  if ($LineX =~/[a-z]$/){chop($LineX);}
83                  $LineNo = $LineX . $TmpLineNo;
84          }else{
85                  $LineNo = $TmpLineNo;
86          }
87          $Text .= $TmpText;
88          push(@LineNums, $LineNo);
89  }
```

```
90
91  sub OutputData{                                  ### 出力ルーチン ###
92          my(@LnTmp, @TextDat);
93          @TextDat = split(/\n/, $Text);
94          @LnTmp = @LineNums; $Line = shift(@LnTmp);
95          foreach $LineDat (@TextDat){
96                  $LineDat =~ s/$CheckPraeCode$CheckPraeCode//g;
97                  $LineDat =~ s/$CheckPostCode$CheckPostCode//g;
98                  $Line = shift(@LnTmp);
99                  if ($OutBufferSize > 0){
100                         &SetupData;
101                 } else{
102                         if (($LineDat =~/$CheckPraeCode/) || $Hit){
103                                 unless ($Hit){
104                                         $OutputLine .= sprintf("%s: %s%5s%s\t%s%s\n",
105                                                         $FileName, $CheckPraeCode,
106                                                         $Line, $CheckPostCode,
107                                                         $LineDat, $CheckPostCode);
108                                 } else {
109                                         $OutputLine .= sprintf("%s: %s%5s\t%s%s\n",
110                                                         $FileName, $CheckPraeCode,
111                                                         $Line, $LineDat, $CheckPostCode);
112                                 }
113                                 $Hit=$True;
114                                 if ($LineDat =~/$CheckPostCode/){
115                                         $Hit=$False; $OutputReady=$True;
116                                 }
117                         }
118                 }
119                 if ($OutputReady){
120                         $OutputLine =~ s/$CheckPostCode/$PostCode/g;
121                         $OutputLine =~ s/$CheckPraeCode/$PraeCode/g;
122                         printf("%s--\n", $OutputLine);
123                         $OutputReady=$False; $OutputLine="";
124                 }
125         }
126 }
127
```

```
128  sub SetupData{                                  ### 整形ルーチン ###
129          if (($LineDat =~/$CheckPraeCode/) || $Hit){
130                  if ($#PraeOutBuffer >= 0){
131                          $OutputLine =
132                                  sprintf("%s\n", join("\n", @PraeOutBuffer));
133                          $#PraeOutBuffer=-1;
134                  } elsif ($#PostOutBuffer >= 0){
135                          $OutputLine .=
136                                  sprintf("%s\n", join("\n", @PostOutBuffer));
137                          $#PostOutBuffer=-1; $PostOn=$False;
138                  }
139                  unless ($Hit){
140                          $OutputLine .= sprintf("%s: %s%5s%s\t%s%s\n",
141                                  $FileName, $CheckPraeCode, $Line,
142                                  $CheckPostCode, $LineDat, $CheckPostCode);
143                  } else {
144                          $OutputLine .= sprintf("%s: %s%5s%s\t%s%s%s\n",
145                                  $FileName, $CheckPraeCode, $Line,
146                                  $CheckPostCode, $CheckPraeCode,
147                                  $LineDat, $CheckPostCode);
148                  }
149                  $Hit=$True; $PraeOn=$False;
150                  if ($LineDat =~/$CheckPostCode/){
151                          $Hit=$False; $PostOn=$True; $HitLast = $Line;
152                  }
153          } elsif ($PraeOn){
154                  if ($#PraeOutBuffer >= $OutBufferSize-1){
155                          shift(@PraeOutBuffer);
156                  }
157                  push(@PraeOutBuffer,
158                          sprintf("%s- %5s\t%s", $FileName, $Line, $LineDat));
159          } elsif ($PostOn && ($WorkedLine < $Line) && ($Line > $HitLast)){
160                  push(@PostOutBuffer,
161                          sprintf("%s- %5s\t%s", $FileName, $Line, $LineDat));
162                  $WorkedLine = $Line;
163                  if (($#PostOutBuffer == $OutBufferSize-1)&&($WorkedLine > $HitLast)){
164                          $OutputLine .=
165                                  sprintf("%s\n", join("\n", @PostOutBuffer));
166                          $#PostOutBuffer=-1;$OutputReady=$True;
167                          $PostOn=$False; $PraeOn=$True;
168                  }
169          }
170          if (eof(IN) && ($OutputLine ne "")){$OutputReady = $True;}
171  }
172
```