テキストファイル版ドイツ語逆引き辞典の作成とその利用

テキストファイル版
ドイツ語逆引き辞典の作成とその利用

城　岡　啓　二
(jjksiro@hss.shizuoka.ac.jp)

0．はじめに

ドイツ語の逆引き辞典は少なくとも最近のものは2冊しかない．東ドイツから出ていたErich Mater（1965）のものは約14万語収録している逆引き辞典だ．辞典といっても見出し語以外に意味やその他の情報があるわけではなく，逆引き配列のたんなる語彙表つまり語彙のリストである．それでも，版を重ね，ドイツ統一以降も他の出版社から出されているところをみると，このての語彙リストは語構成などの研究に利用されているようで，研究者の需要があったようである．西ドイツから80年代の後半に出たGustav Muthmann（1988）の逆引き辞典は後発ということもあって，語数が17万5千語でMaterのものより

多く，配列も文字の完全な逆引き配列ではなく，発音を考えて場合によっては配列をずらすなどして新しい工夫をした逆引き辞典である．

したがって，すでに2冊のドイツ語の逆引き辞典があるわけだが，どちらも紙の辞典でパソコンを利用した高度な情報処理には使えない．テキストファイル版ドイツ語逆引き辞典があればgrepなどの文字列検索ツールが使え，検索が容易だというだけでなく，従来の紙の逆引き辞典では不可能な語中の要素の検索もおこなえる．語中要素の研究はこれまでは記憶と内省をもとにするほかなかったわけだが，正規表現という記法の使えるgrepなら「..*licher..*」とすれば，語中に-licher-を含む語を語彙リストから正確に素早く抜き出すことが可能である．こういうテキストファイル版の逆引き辞典がいくら便利だといっても語構成についてしらべたいというひとは言語学者か語学教師にふつうは限られるわけで，需要が大きいとは言えない．そのためだと思うが，テキストファイル版の逆引き辞典はドイツにおいてもまだ発表されていない．ないなら自作してしまえということで作成したのだが，これが可能になったのは，まず，Sonyのつくった電子ブックと電子ブックをパソコン上で扱うフリーソフトDDwinのおかげである．また，編集作業において必要不可欠な役割を果たしたのがsortやuniqやrevやawkやsedといった汎用テキスト処理ツールである．

電子ブック（Data Disc）という規格で多くの辞典類が発売されている．広辞苑や大辞林や数々の英和辞典など日本製の辞書・事典類だけでなく，Concise Oxford DictionaryやAmerican Heritage Dictionaryなど海外のものもある．ドイツ製の電子ブックはドイツ国内では普及せず既に入手不可能になったようだが，過去にはかなりの辞典類が発売されている．現在でも日本ではDuden UniversalwörterbuchやMackensenのDeutsche Rechtschreibungなどが電子ブック版で入手可能である．これらの電子ブック版の辞典類は検索のスピードが早いし，電子ブックによっては条件検索といって見出し語ではなく解説文中のキーワードにしたがって検索することも可能で（たとえば広辞苑の電子ブックで「ほうげん」と「がいこくご」の条件検索をするとこのふたつを解説文中にふくむ「借用語」が検索できる），このままでも十分に利用価値のあるものであるが，中身の情報をテキストファイルで取り出すことができれば，たんなる辞書の検索ではないような語学研究にも使えそうである．

DDwinというのは，電子ブックやEPWING規約の12cmCD-ROMをWindows上（現在は3.1用と95用がある）で使えるようにする草本和馬さんのつくったフリーウェアで，パソコン通信やパソコン雑誌の付録などで最新版が手

に入る．また，NIFTY-Serve で作者自身がサポートしている．DDwin はヴァージョンアップのたびに機能が追加され，索引情報によらない全文検索（条件検索と似ているが，条件検索は索引情報が電子ブック内に記録されている場合にだけ可能になる）など，元の電子ブックプレーヤでは不可能な検索もできるようになっている．1.48 では電子辞書の見出し語だけ，あるいは，見出し語と解説文の両方を一括してテキストファイルとして取り出す機能が加わった．しらべてみると，この機能を利用すれば，一枚のディスクに入ったすべての見出し語でも効率的に取り出せることが分かった．

また，現在，各種の汎用テキスト処理ツール(UNIX-like Tools とか UNIX-like Utilities とか Text Tools あるいは DOS Tools などと称していることが多い）がインターネットやパソコン通信であるいは CD-ROM を媒体に出回っているが，これらの汎用ツールを使うだけで一行一語の語彙リストを逆引き配列にすることができることが試行錯誤の結果分かった．逆引き配列にする際に必要なのは rev と sort で，重複したデータを整理するのが uniq だ．また，rev のかわりに awk を使うこともできる．不要なデータの削除や一括変換には sed が便利だ．また，cut, paste, fold, dd, nl, tr, comm, wc なども含め UNIX のコマンドに由来するツール群はテキストデータを扱うならぜひ使い方を覚えたい便利な道具である．

以下では，電子ブックから見出し語を効率的に取り出す方法，取り出した見出し語データを逆引き配列で整理する方法，テキストファイル版逆引き辞典の編集上の留意点と利用法について，後発の研究者の参考になるようにできるだけ具体的に記述しておきたいと思う．記述に一貫性をもたせるため電子ブック版クラウン独和辞典の場合を例としてあげることが多いが，他の電子ブックからでも基本的には同様の方法で見出し語データは取り出せ，テキストファイル版逆引き辞典が作成できる．

1．電子ブックの見出し語データを取り出して，整理する

1．1 DDwin で見出し語を取り出す

DDwin を利用して電子ブックから見出し語データを取りだすのであるが，こういう使い方は語学研究者でなければ不要な使い方で，一般のひとには思いもよらぬ作業だろう．また，DDwin のふつうの使い方でもない．したがって，DDwin の詳細なヘルプファイルを熟読してもこれから述べるようなことは書

かれていない．DDwin は基本的には電子ブック検索ソフトウェアであって，見出し語や辞書の中身をテキストファイルで取り出すためのソフトウェアではないからだ．

見出し語データを DDwin で取り出す方法を順を追って説明していこう．

a．まず，DDwin を動かしはじめる前にいくつか設定を確認しておく．メニューバーの「その他」を選択し，さらに「環境設定」を選択し，表示されるダイアログボックスで「高度」を選択する．これで「高度な設定」のダイアログボックスが表示されるのだが，「上級者メニュー」が選択されていなかったら選択する．また，「全項目自動表示」が選択してあれば選択をはずしておく．これが選択してあると，見出し語をテキストファイルに取り出すのに不要な画面表示のために時間を取られることになるし，場合によっては，扱う量が許容範囲を越えるため，「表示可能な行数を越えました．超過行を無視します」という表示がでて，見出し語をすべて取り出すという目的が達成できないので要注意である．それから「編集ファイル名」には保存したいディレクトリ名ぐらいまでは書いておいたほうがあとで入力する手間がはぶけて便利である．

b．電子ブックを CD-ROM ドライブに入れ，「文献」を選ぶ．「外字の扱い」を「16 進表示」に設定する．これをしないと，テキストファイルに出力する際にドイツ語の特殊文字が奇妙な文字に変換されてしまうためである．「外字の扱い」が表示されない場合は「上級者メニュー」になっていないので，メニューバーの「その他」から「環境設定」－「高度な設定」で「上級者メニュー」に設定しなおす．

c．検索の準備は終わったので，いよいよ検索する．ワイルドカードのアステリスクを使って，「*a」または，「a*」を「索引検索」で検索させる．アステスクをどちらに付けるかは最初に選択してしまおう．「*a」なら a で終わる見出し語を逆引き辞典の配列で，「a*」なら a で始まる見出し語がふつうの辞書の順番で取り出すことになるのだが，どちらを選んでも基本的には見出し語はすべて取り出せるし，見出し語さえ取り出してしまえば，後述するように逆引き配列にすることはどちらの場合も可能である．逆引き配列で見出し語を取り出す場合は，DDwin の設定で「高度な設定」の「該当リストのソート」

のチェックを外しておこう．この方法で取り出した見出し語データをcopyやcatで順番につなぐだけでひとつの電子ブック版辞書からなら逆引き辞典がつくれる．しかし，複数の電子ブック版辞書の見出し語を結合して編集する場合には次節で述べる方法で見出し語の整理や配列の変換が必要である．「候補表示-選択エリア」を見ているとすこし時間がかかるが，そのうちに見出し語が連続して表示される．見出し語が「候補表示-選択エリア」に表示されたらこのエリアの下部にある「全て」というボタンを押して，検索された見出し語をすべて選択状態にする．それから，メニューバーの「編集」から「エディター起動」を選択する．次に出てくる表示で出力したいファイル名を指定して，「該当項目見出しのみ」をチェックする．そして，「OKボタン」を押せば，あとはDDwinが登録してあるエディタを使って自動的にテキストファイルに見出し語を出力してくれる．エディタはWindowsに付属のものでは大きなファイルが扱えないので，巨大ファイルでも扱える別のテキストエディタを用意しておく必要がある．なお，「索引検索」でなく「全文検索」を利用しても見出し語が取り出せる場合がある．[1)]

d．「*b」から「*z」まで(c)の作業を繰り返す．電子ブック版のクラウン独和辞典ではウムラウトは指定しなくても検索できるようになっているので「*ö」などとする必要はない．またエスツェットで終わる語も「*s」で検索することになっている．ドイツで発売された電子ブック版ドイツ語辞典ではウムラウトやエスツェットを入力する必要があるが，日本語版Windows 3.1ではコード入力することになる．なお，電子ブックでは大文字と小文字は内部では無視してしまうため，検索の際にはどちらを使っても結果は同じである．ドイツ語のすべての特殊文字がこのコード入力でうまく行くかどうか私自身は確認していない．私自身は英語版Windows 3.1で入力している．DDwinは表示をすべて英語に切り換えることができ，そのまま英語版Windowsで使用することができる．

e．(c)と(d)の作業でつくった26個のファイルを連結する．DOSのcopyコマンドでこれができる．26個のファイルをdaten.a，daten.b，daten.cというふうにつくっていれば，「copy daten.* daten.dat」とすれば，最終的に26個のファイルが結合され，daten.datというファイルができあがる．

ここではテキストファイル版の逆引き辞典の作成との関連でDDwinによる見出し語の抽出法について述べたが,上記の(c)の工程で「該当項目見出しのみ」のチェックをしなければ辞書のすべてがテキストファイルとして取り出せる（ただし，発音記号などは文字化けしてしまう）．私は実際Dudenの Universalwörterbuchでそうしている．こういう使い方ができるのはDDwinのおかげであり，現時点ではDDwin以外ではちょっと実現が難しいのではないかと思われる.[2)]

1.2 取り出した見出し語の後処理

[1.1] の作業でできたファイルを完全な一行一語の語彙リストにするには，まだかなりの加工が必要だ．とはいえ，作業の大部分はエディタソフトの一括置き換えや一括削除をつかえばよいので，それほどの作業ではない．まず，見出し語以外の不要な記号を削除しなければならない．クラウン独和辞典の例だと最重要語の前にはアステリスクふたつが縦に並んでいる外字が，重要語には上付きのアステリスクひとつが付加されている．「外字の扱い」が「16進表示」にしてあればテキストファイル中では [a424] と [a423] となって出力されているはずだ．[a424] や [a423] をエディタなどで一括削除すればよい．また，綴りの分け方を示すのにナカグロがあるから，これも一括で削除．また，複合語内部の要素間の切れ目を示す語構成表示のための記号としてイコール記号を波線にしたようなもの（[a16 f]）が使われているが，これも一括削除した．それから，見出し語に続けて解説文の一部の記号などが出力されるケースもあったが，これも削除である．分離動詞の前つづりと動詞本体のあいだにはいる下向きの矢印（[a171]）も削除した．クラウン独和辞典の見出し語データの整理で一番面倒だったのが,複数の見出し語が一行に出力されたものの処理である．たとえばNesselausschlag, Nesselfieber, Nesselsuchtはひとつの見出しとして扱われているのである．基本的にはこれを3行に切り分ければいいのだが，やっかいなのは，3語がひとつの見出しとして扱われていることでDDwinの見出し語表示限界(電子ブックプレーヤでも表示しきれない場合がある)を越えてしまうようで見出し語全体が取り出せないことである．3番めのNesselsuchtでは実際に出力されたのはNだけだった．こういう場合は出力されなかった語が何であるかもう一度個別に検索して確認する作業が必要になって，かなり時間をとられる．また，そもそも見出し語だけを取り出すことができないような電子ブックもある．見出し語といっしょに解説文の一部が出力されて

しまうのだ．こういう電子ブックから見出し語だけを取り出したい場合は，解説文の一部を取り除く作業が必要になる．

ここまでの作業でかなり整理されてきているが，さらに，次の方針で一行一語の語彙リストの編集を続けた．

a．( )は使わない．
b．一行一語にして行末のスペースは削除する．

Selbstbestimmung(srecht) のような書き方は人間が直接読むには問題ないが，パソコンで検索することを考えると，Selbstbestimmung と Selbstbestimmungsrecht の2語にしなければならない．単語のあとにスペースがある場合は，これがあると，grep などで検索するのが面倒になる．たとえば，合成語の語中に出てくる ung を検索しようとして「..*ung..*」という正規表現（[3.4]，[4.1] 参照）を使うと，スペースも文字の一種と見なされ，「Begegnung スペース」のようなものが意に反して検索されてしまう．また，語末にスペースがあると，逆引き配列にするために rev を使って行を反転させたあと sort するとスペースもコンピュータの処理としては文字であるから，こちらの期待通りの配列にならない．

なお，人間の行うことはどうしても間違いが入り込む可能性がある．電子ブック版の辞書は中身が紙と違って見えないだけに，妙な間違いが編集過程ではいったりしても十分にチェックされずにそのまま出版されてしまう危険性が大きいようだ．たとえば，drawer が dr0wer，interline が in0erline になっていたり，Dominante や Konstatierung が Domi nante や Konsta tierung と出力されたりする電子ブックがあった．この種の明らかな間違いは気付いた範囲で直した．ちょっと驚かされたのは，電子ブック版の Hermann Paul: Deutsches Wörterbuch の不具合で，この辞書では索引情報のつけ方にミスがあったようで，前方一致検索では検索できない語がかなりあった．たとえば，ふつうの前方一致検索では sein は検索できず，後方一致検索でないと検索できなかった．したがって，この辞書では前方一致検索ではすべての見出し語は取り出せない．ハイフンについても問題はある．まず，出力されたハイフンは形式が不統一だったり，キーボードから直接入力できないものだったりした．これはすべてキーボードにあるふつうのハイフンに一括置き換えした．さらに，ハイフンでは，単語が行末に来たためにはいったハイフンも Al-tersruhegeld のように出力されてしまった．こういうハイフンは面倒でも手作業で削除していかなければな

らない．編集方針によってはさらに各種の削除が必要になるが，これについては後で述べる．

2．一行一語の語彙リストを逆引き配列にする

rev というテキスト処理ツールがある．UNIX の種類によっては最初から備わっているコマンドであるが，MS-DOS 用として提供されているものは1種類しか発見できなかった．まあ，後で述べるように，rev の機能は awk でも可能だから rev が必要不可欠というわけでもない．rev はどういうツールかというと，テキストファイルの各行の内容を反転するツールである．一行一語の語彙リストを逆引き配列にするには，基本的には rev と sort を組み合わせるだけでよい．例として sportler.0 という名前のテキストデータをとりあげよう．内容は次の7行とする．様々な Sportler がふつうの辞書と同じ順番で並んでいる．各種の Sportler のあいだには無関係の語（Deutsch, Rathaus, Splitter）がわざと入れてある．

```
Berufssportler
Deutsch
Profisportler
Rathaus
Spitzensportler
Splitter
Sportler
```

rev で sportler.0 の内容を書きかえて sportler.1 というファイルに保存する．

```
rev sportler.0 > sportler.1
```

sportler.1 の内容は次のようになる．

```
reltropssfureB
hcstueD
reltropsiforP
suahtaR
reltropsneztipS
```

rettilpS
reltropS

ここで sportler.1 の各行をソートすると，行の順番が入れ代わる．語頭からソートしていくことになるが，本来は語末であるものが語頭に来ているわけだから，語末の文字から語頭にかけて語彙をソートしていくという意味になる．sort は各種のものがあって，使い方も多少ことなるのだが，私の使ったものでは次のようにキーボードから打ち込む．なお，ドイツ語で逆引き辞典をつくる場合には大文字と小文字の区別をしないいわゆる辞書順のソートがどうしても必要なのだが，この例ではとりあえず無視している．

sort sportler.1 > sportler.2

sportler.2 の内容は次のようになる．

hcstueD
reltropS
reltropsiforP
reltropsneztipS
reltropssfureB
rettilpS
suahtaR

これをあらためて rev で反転させると配列を除けば元どおりの単語に戻って，逆引き配列の語彙リストが完成する．

rev sportler.2 > sportler.3

sportler.3 の内容は，

Deutsch
Sportler
Profisportler
Spitzensportler
Berufssportler
Splitter
Rathaus

のようになっていて，各種の Sportler がならんで出現する語尾部分のそろった逆引き配列になっていることが確認できる．

ここで述べた方法で語彙リストを逆引き配列するには，行の内容を反転させる作業がどうしても必要である．しかし，rev は一般に普及しているツールとは言い難い．UNIX でもシステムによっては rev コマンドがないし，MS-DOS 上で使える rev が入手できない場合もあるだろう．そういう場合は，上ですでに述べたように awk で rev の機能が実現できる．R. Stallman 他 (1993, p. 109) や植村・富永 (1993, pp. 119-120) にスクリプトがある．私もつくってみたが，ごく短いスクリプトで実現できる．

```
# REV.AWK
  {for (i=length () ; i>0 ; i--)
     printf "%s", substr ($0, i, 1)
   print ""
  }
```

sportler.0 の各行の内容を反転させ，その結果を sportler.1 に保存するには次のようにする．

```
awk -f rev.awk sportler.0 > sportler.1
```

rev や rev.awk はテキストファイルの語彙リストならなんでも適用可能である．要は一行一語の語彙リストを入手することであるが，電子ブック版の辞書からなら [1] で述べたように DDwin を利用して見出し語だけを取り出すことができるし，語学教材を対象に使用語彙を逆引き配列にするぐらいなら量は多くないので手作業で入力するのもいいだろう．また，スキャナと OCR ソフトでまずテキストを入力して，それを一行一語の語彙リストに word などのツールを使って処理する方法もある．また，将来的には一行一語のテキストファイルの語彙リストが流通する可能性もあるだろう．実際，インターネットを探すと，英語の語彙リストなら未整理だと断りのついている語彙リストであるが，現在でも 10 万語程度のものは入手できるようだ．

ツールの rev にしても awk スクリプト rev.awk にしても行全体を反転する仕組みになっていて，行の第 1 フィールドだけ反転させたいという場合には使えない．しかし，たとえば第 2 フィールド以降に説明などの付加情報のはいっ

ている語彙リストを逆引き配列にしたい場合もあるだろう．こういう場合にはrevではどうしようもないが，awkならスクリプトに手を加えることで対応できる．

```
# REV2.AWK（各レコードの第1フィールドだけ反転させる）
  {nagasa=length($1)
   for (i=nagasa ; i>0 ; i--)
     printf "%s", substr ($1, i, 1)
   print substr ($0, nagasa+1)
  }
```

たとえば次のような付加情報のはいった語彙リストのファイルがあるとする．ファイルの名前はfile.0としておこう．

```
Ferienhaus          Nr. 1
Ferienwohnung       Nr. 2
Hauswesen           Nr. 3
Sommerhaus          Nr. 4
Sommerwohnung       Nr. 5
Wohnungswesen       Nr. 6
```

ふつうの辞書の順序にならんでおり，語頭の部分が共通している語も混じっている．付加情報の例として番号を書き加えてみた．さて，rev2.awkをつかって逆引き配列に並べかえるには次のように打ち込む．

```
awk -f rev2.awk file.0 | sort | awk -f rev2.awk > file.1
```

file.1という新しいファイルができるが，その内容は，

```
Ferienwohnung       Nr. 2
Sommerwohnung       Nr. 5
Wohnungswesen       Nr. 6
Hauswesen           Nr. 3
Ferienhaus          Nr. 1
Sommerhaus          Nr. 4
```

となっていて，語彙の出だしはふぞろいになってしまったが，後半部がそろっ

た並び方にふたつずつなっている．要するに，逆引き辞典の配列に正しく変換されている．行の後半も正しく移動していることは数字の対応を見ればあきらかだろう．たとえばNr. 1がFerienhausだという事実は変化を受けていない．

付加情報つきの語彙リストが逆引き配列にできればどんなメリットがあるだろうか．逆引き辞典の作成と直接関係はないが，具体例をひとつあげておこう．動植物名の一部に形容詞が使われているものがあるが，これを整理してみたことがある．まず，Harry Garms (1969) : Pflanzen und Tiere Europas. München (dtv). の索引から形容詞つきの動植物名をすべてテキストデータとして入力した．入力形式は形容詞の活用をとった代表形をまずあげ，それから形容詞つきの動植物名とした．ウムラウトは¥記号をaやoやuの前に付加し，エスツェットもsの前に¥記号を付加することであらわしている．

schwarz	Schwarzer Aask¥afer
abgestutzt	Abgestutzte Klaffmuschel
achtf¥u¥sig	Achtf¥u¥sige Zweikiemer

こんなふうに入力した行がえんえんと続くわけだが，たとえば色彩のrotに着目して，rot-で始まる合成形容詞の例が知りたいとする．これはファイルを単純にsortで並べかえればよい．

rotadrig	Rotadrige Singzikade
rotbauchig	Rotbauchige Unke
rotbeerig	Rotbeerige Zaunr¥ube
rotbraun	Rotbraunes Kopfried
rotgeb¥andert	Rotgeb¥anderter Totengr¥aber

こういう行が簡単に見つかる．それでは，今度は，-rotで終わる合成形容詞が知りたい場合はどうすればよいか．行の先頭にある形容詞を逆引き配列にすればよい．rev2.awkを利用してそういうファイルをつくると，

orangerot	Orangerotes Habichtskraut
fleischrot	Fleischrotes L¥ausekraut
ziegelrot	Ziegelroter Ri¥spilz
ziegelrot	Ziegelroter Schlangenstern
rostrot	Rostrote Segge

rostrot	Rostrote Uferschnepfe
blutrot	Blutroter Schnellk¥afer
blutrot	Blutroter Seeampfer
blutrot	Blutroter Storchschnabel

こういう調査が簡単にできるわけだ[3)]

逆引き配列は単語の逆順配列と行のソートというふたつの組み合わせで可能になるわけだが，ソートについてはまだ触れていない問題点がある．

a．大きなファイルでもソートできるか？
b．辞書順のソートが可能か？

MS-DOS にも sort.exe がついている．しかし，これだとちょっと大きなテキストファイルがもう扱えない．扱えるファイルの大きさはマニュアルで 64 KB までとなっている．実際にどのぐらいの大きさの語彙リストなら扱えるかしらべてみた．一行一語の語彙リストで確認すると 7500 行程度でもうソート不能になった．これではとてもテキストファイル版逆引き辞典の編集には使えない．さいわい，sort は MS-DOS で使えるものが多く出回っていて，大きなファイルが扱えるものも多い．テキストファイル版逆引き辞典の作成では，既存の電子ブックの見出し語をもとにつくるならば一行一語のテキストファイルが 20 万行程度扱えればよい．ただ，辞書順のソートも可能でないと困る．本来，sort はコード順に行を並びかえるものらしいのだが，このコード順というのは辞書の順番と大文字と小文字の区別という点が大きく異なる．コード番号は大文字のほうが小さく，ABC...XYZabc...xyz という順番である．だから，この順番だと，すべての大文字で始まる語が先で，その後に小文字で始まる語が続くことになる．名詞を大文字で書くドイツ語ではとても容認できない順番になってしまう．例をあげて説明しておこう．

essen
Essen
ewig
Ewigkeit
Floskel
Flo¥s
Flosse

これは辞書の順番にならべたものだ．UNIX の sort をもとにつくられ，UNIX-like Tools という名前で配布されている sort.exe で上の内容のファイルをオプションをつけずにソートした結果が次のものである．大文字で始まる Flosse が小文字で始まる essen の前に来ている．

Essen
Ewigkeit
Flo¥s
Floskel
Flosse
essen
ewig

今度は，大文字と小文字を区別しない-f オプションをつけてソートしてみる．

Essen
essen
ewig
Ewigkeit
Floskel
Flosse
Flo¥s

となって，辞書の順番に近いが，Essen が essen の前に来ている点がもとの辞書の順番とことなる．つまり，大文字と小文字の区別をしないが同順位になる場合は大文字を小文字にを優先させているわけだ．また，エスツェットを¥s であらわしていることもあってこの順番も本来の順番からずれてしまうが，これはやむを得ない．なお，私がテキストファイル版逆引き辞典の編集に利用したのは xsort という名称のソート・ツールだが，これだと，essen を Essen の前に出すソートが可能だ．

3．テキストファイル版逆引き辞典をつくる

編集作業の諸問題を［3.1］から［3.4］で述べる．

3．1 結合した複数データの整理

辞書一冊の見出し語データでも同音語，正確に言えは，同字語がかなりある．これをこのままにしておくと，コンピュータで各種のテキスト処理する場合に妨げになる．現状では同一の見出し語はひとつだけにしておくのがよさそうである．複数の辞書からの見出し語データをcopyやcatで結合した場合，当然，かなりの部分は見出し語が重複しているはずだ．数万から数十万あるファイルで重複行を手作業で削除していくことはとてもできない．こういうときのために汎用のテキスト処理ツールsortとuniqが使える．

sort［ファイル］｜uniq ＞［新ファイル］

UNIXのsortやこれの移植版だったら-uオプションがあって，これにはuniqの複数の同一行を一行にまとめる機能が含まれているから

sort -u［ファイル］＞［新ファイル］

とすればよい．

3．2 Umlautなどの特殊文字の処理

ウムラウトやエスツェットについては，基本的には日本独文学会データベース委員会の方法で「ドイツ語特殊文字は，¥記号を当該文字の直前に付加する［例：¥A；¥a；¥s；¥o；¥O；¥U；¥u］」というエスケイプ2文字方式を採用した．ただし，この方式には問題もあるので，研究内容によっては，未使用記号方式でもファイルをつくって利用することにした．

エスケイプ2文字方式の問題点は，語彙の文字数を問題にしたりする場合は本来1字のものを2字で表現しているため不正確になるか，処理が複雑になってしまうということである．また，特定の文字や文字列の位置を文字数でかぞえるような場合にもエスケイプ文字の¥記号などの分だけ字数が増え位置もずれる．たとえば，Flu¥sは本当は4文字なのに¥のせいで5文字になってしまう．W¥aschekorbにはascheなど本当は含まれていないのに含まれているこ

とになってしまう．Fl¥ache の ch は a に後続しているわけではないのに後続していることになる．¥Arger の１字めは本来大文字なのにこの書き方では２文字めが大文字である．

エスケイプ２文字方式の問題点のほとんどは，特殊文字１字を別の未使用の記号１字に割り当てれば解決する．利用できる記号は語彙リスト中に未使用の記号でたとえば「¥s」のかわりに「*」を「¥a」のかわりに「＋」を「¥A」のかわりに「#」を利用すればよい（sed のスクリプトが読めるひとは［3.4］の記述を読めば私が未使用記号方式で利用した記号の詳細が分かる）．未使用記号方式で特殊文字を記述すると，可読性が低く，人間が読むには適していないが，パソコンで処理する際にだけ使うことにすればよい．また，「¥a」のかわりに「*」を使うような未使用記号方式には可読性以外にもデメリットがあるのでまとめておこう．

a．未使用記号の数は限られているから，表現できる特殊文字が少ない．
b．ソートするとその部分がふつうの辞書の順序とずれてしまう．
c．記号を使うので大文字と小文字の変換ができなくなる．

(c)について補足すると，ドイツ語では名詞を大文字で書くので，大文字と小文字の変換という作業が必要になることがけっこうありそうだ．たとえば，文字の頻度を出したいのだが，大文字と小文字を区別したくないということもあるだろう．こういうときには未使用記号方式は無力である．大文字と小文字の変換をしてくれるツールは多い．しかし，awk の tolower()関数や toupper()関数や dd の conv＝lcase や conv＝ucase で大文字と小文字の変換が可能なのはアルファベットだけである．「¥a」のかわりに「＋」と書いてしまえば，「＋」は大文字でも小文字でもないから変換の対象にはならない．

さて，エスケイプ２文字方式でウムラウトなどの特殊文字を記述する際には，ドイツ語以外の特殊文字についても置き換えのルールを定める必要がある．ドイツ語の逆引き辞典をつくる場合に限っても，見出し語のなかにはフランス語やチェコ語のものを始めかなり多くの外来語などが混じっていて，それぞれの国の特殊文字が使われていることもあるからだ．データそのものから除外してしまうことも考えたが，Café のように重要な語彙も混じっているからすべてというわけにもいかない，それで，一応，データベース委員会方式を拡張した置き換えのルールを定めることにした．なお，中にはドイツ語の辞典には見られないスペイン語の文字も混じっているが，これは英語（米語）の逆引き辞典を

編集する際に頻繁に出てくるスペイン語の文字に対応したものである．

(1) i の上に¨がついた文字 (ï) ……¥i
(2) e の上に¨がついた文字 (ë) ……¥e
(3) セディーユ (小文字) (ç) ……¥c
(4) セディーユ (大文字) (Ç) ……¥C
(5) o と e の合字 (œ) ……&o
(6) O と E の合字 (Œ) ……&O
(7) n に波線 (˜) がついた文字 (ñ) ……¥n
(8) a に右下がりのアクサン・グラーヴ (à) ……>a
(9) a に左下がりのアクサン・テギュ (á) ……<a
(10) e に右下がりのアクサン・グラーヴ (è) ……>e
(11) e に左下がりのアクサン・テギュ (é) ……<e
(12) u に右下がりのアクサン・グラーヴ (ù) ……>u
(13) u に左下がりのアクサン・テギュ (ú) ……<u
(14) a にアクサン・スィルコンフレックス (â) ……^a
(15) e にアクサン・スィルコンフレックス (ê) ……^e
(16) i にアクサン・スィルコンフレックス (î) ……^i
(17) o にアクサン・スィルコンフレックス (ô) ……^o
(18) u にアクサン・スィルコンフレックス (û) ……^u
(19) c にハーチェク (č) ……&c
(20) C にハーチェク (Č) ……&C

3.3 見出し語として採用する語および語形

すでに一行一語の語彙リストという形式と行末にはスペースを置かないという点，また，語彙を記載する際には括弧は使わずに異形は 2 語として記載すること，見出し語以外のデータ（重要語の印，不規則変化動詞の印，分離動詞の分離線，分綴の印など）は削除することを [1.2] のところで述べた．編集作業をさらにすすめるには，最終的に採用する見出し語を決めて，それに応じて削除や書き換えが必要である．どういう研究に利用するかでテキストファイル版逆引き辞典に採用する見出し語は変わってくるはずだが，私自身は一応次のような編集方針をとにかく決めた．

a．同音意義語の配慮はしない．したがって，一語形は一度だけ記載する．分離動詞と非分離動詞の区別もしない．
b．見出し語は単語だけとする．したがって，文字や略語や記号や単位などは採用しない．それに接頭辞や接尾辞も載せない．また，2語以上からなる表現(たとえば諺)も削除する．2語以上のものはフィールド単位でテキスト処理する際に例外的な扱いが必要になる．定冠詞つきで名詞が見出し語になっている場合は定冠詞を削除する．
c．動詞は不定詞しか載せない．過去分詞は形容詞化したものだけ残す．
d．形容詞は活用語尾の付かない形しか載せない．
e．地名や国名などの固有名詞は載せる．Sri Lanka や New York のように2語からできている地名や国名は削除した（(b)を参照)．人名は家族名は載せないことにした．Liszt や Nietzsche や Lukács などの有名人の名前は辞書の見出し語になっていたりするが，ドイツ語の語彙として残すことには問題があると判断したからだ．ただし，Monika や Klaus などの個人名は削除しなかった．

一語形は一度だけ記載するというのは，動詞の sein と所有代名詞の sein も区別しないということで問題のあるやり方であるが，テキスト処理を手軽に行うには必要である．[3.1]で述べた uniq をつかった重複行の処理などでもこの前提があってはじめて可能になる．

今のところまだ編集上の方針が決められなくて，元のデータをそのままにしている点もある．

a．複数形の見出し語をどうするか？
b．冠詞類や代名詞の語形をすべて採用するかどうか？
c．形容詞型変化の名詞の見出し語としての語形をどうするか？
d．序数詞の見出し語としての語形をどうするか？
e．形容詞の比較級や最上級の見出し語を削除するかどうか？
f．数字や分数や回数をどれだけ載せるか？

こういう問題がなかなかやっかいだということを複数形の見出し語で考えておこう．複数形しかない Eltern のような場合には複数形の見出し語を採用するしかない．それでは単数形がないわけではないが，ふつうは複数形が使われる語についてはどうするべきだろうか．Streitigkeit と Streitigkeiten のような場合

である．クラウン独和辞典では「ふつう複数で」としながらも見出し語に採用しているのは単数形だ．しかし，辞典によってはこういう場合に見出し語として複数形を採用しているものもある．また，クラウン独和辞典は学習辞典だから重要語については Häuser のように複数形も見出し語としているものがかなりあるようだ．かりに複数形の扱いが決まったとしても，書き足すのも削除するのも辞書全体を手作業で確認しながら行うことになり，たいへんな作業となる．

3．4 sed による一括削除，一括変換[4)]

編集作業においては一定パターンの一括削除を繰り返し行うことが多い．削除するリストをつくれば繰り返し活用できる sed が便利である．また，sed を利用することで変換と復元という作業を自動化することもできる．一括削除の例として接頭辞と接尾辞の削除の例をあげておこう．接頭辞や接尾辞の行をすべて削除するスクリプトは次の2行でいい．

```
/^-/d
/-$/d
```

/^-/や/-$/は妙な記号の羅列に見えるかもしれないが，正規表現と言われる書き方（[4.1] 参照）で汎用性があり，覚えてしまえば，他の awk や grep などでも同じように使える．上の2行のスクリプトの内容は，「行頭にハイフンがある行と行末にハイフンがある行を削除しなさい」という意味になっている．これは sed による一括削除が非常にうまく行く場合だが，場合によっては削除リストの作成にかなりの作業を覚悟しなければならない．電子ブック版クラウン独和辞典では紙の辞典なら巻末についている動詞変化形索引がふつうの見出し語といっしょに取り出されたため動詞の変化形の削除を sed でおこなった．この場合は数千語を手作業で入力して削除用スクリプトを作ることになった．しかし，この場合でも，他の電子ブック版の辞書から取り出したデータに対してもこのスクリプトが使えるし，失敗があって元のデータから編集作業をやり直したい場合も大助かりである．

[3.2] で述べたようにウムラウトなどの特殊文字は，¥などの記号をつかって基本的にはエスケイプ2文字方式であらわすことにしている．ただし，語彙の文字数の調査などの場合は必要に応じて未使用記号方式で特殊文字を表わすファイルを利用するということも述べた．エスツェットならエスケイプ2文字

方式では「¥s」だが，未使用記号方式では「*」ひとつであらわしている．テキストファイル版辞書の編集作業では，ふたつのファイルで別々に編集作業をすすめるというのは効率的ではない．だから，編集作業などはエスケイプ２文字方式のファイルに対して行い，未使用記号方式のファイルが必要ならエスケイプ２文字方式のファイルを変換してつくることにした．そのため，編集の各段階でくり返しエスケイプ２文字方式から未使用記号方式に変換するという作業を行なっている．sed ならこういう一括変換はスクリプトをファイルに保存しておけるので便利である．また，未使用記号方式のファイルで語彙の文字数の処理などをしてその結果に未使用記号方式の特殊文字が含まれている場合は，その結果を再び見やすい¥などをつかったエスケイプ２文字方式の特殊文字に復元することもできる．次に私が実際に変換と復元の目的で使用しているスクリプトをあげる．実際に使用する sed のスクリプトは「s///g」の形式で一行にひとつずつ書くのだが，それでは場所をとるので，ここでは１行に４行分を書いておくことにする．

ａ．エスケイプ２文字方式の特殊文字を未使用記号方式に変換する：

```
s/¥¥s/*/g    s/¥¥a/+/g    s/¥¥A/#/g    s/¥¥o/$/g
s/¥¥O/%/g    s/¥¥u/~/g    s/¥¥U/_/g    s/<e/(/g
s/>e/)/g     s/<a/[/g     s/>a/]/g     s/¥¥n/=/g
s/¥^a/:/g    s/¥^o/;/g    s/¥¥c/@/g    s/¥¥i/0/g
s/¥¥e/0/g    s/¥¥C/0/g    s/<u/0/g     s/>u/0/g
s/¥^e/0/g    s/¥^i/0/g    s/¥^u/0/g    s/&o/"/g
s/&O/!/g     s/&c/0/g     s/&C/0/g
```

ｂ．未使用記号方式の特殊文字を元のエスケイプ２文字方式に復元する：

```
s/¥*/¥¥s/g   s/¥+/¥¥a/g   s/#/¥¥A/g    s/¥$/¥¥o/g
s/%/¥¥O/g    s/~/¥¥u/g    s/_/¥¥U/g    s/(/<e/g
s/)/>e/g     s/¥[/<a/g    s/¥]/>a/g    s/=/¥¥n/g
s/:/^a/g     s/;/^o/g     s/@/¥¥c/g    s/"/¥&o/g
s/!/¥&O/g
```

復元するほうのスクリプトの量がはるかに少ないのは復元をあきらめているものがかなりあるためである．大文字のセディーユやハーチェクなどはエスケイ

プ2文字方式で記述するには問題はないが，未使用記号方式で別の記号に割り当てるには記号が足りない．それで，頻度の少ない特殊文字はすべて「0」に置き換えている．語彙の文字数や文字位置を問題にするような場合，これらの文字自身が関係しなければこれで十分である．しかし，当然，「0」に置き換えてしまったものは元の記号には戻せない．1から9の数字に割り当てることも不可能ではないが，場合によってはこの辺りのアラビア数字が語彙の一部に使われていることもあって，問題がないわけではない．特殊文字の変換および復元スクリプトで注意しなければならないのは，よく考えて変換先の記号を決めないと予想しない結果になるということである．細心の注意と試行錯誤が必要である．たとえば，「¥u」の変換先に「&」を指定したりすると，B¥ucher がまずB&cher に変換され，スクリプトの後ろのほうでふたたび「&c」が「0」に変換されて，B0her が生成されてしまう．だから，置き換え文字を決める際には変換先と変換元の両方を考慮にいれなければならない．

4．テキストファイル版逆引き辞典を利用する

現時点で編集が一番すすんでいるのは，クラウン独和辞典と Langenscheidts Data Disc Wörterbuch Französisch と Langenscheidts Eurowörterbuch Italienisch の3冊の電子ブックから取り出したドイツ語の見出し語データをもとにつくったテキストファイル版逆引き辞典だ．一応 jisho.dat と名付けておこう．jisho.dat は今後整理がすすめばさらに小さくなる可能性があるが，1996年4月7日現在で語数が55464語で平均文字数が10.0337字になっている．一行一語のデータだから語数は行数であり，各行の文字数をどんどん足していって，その総和を総行数で割れば平均文字数がでる．これを awk でやるなら

```
awk "{all+=length($0)}END{print NR, all/NR}" jisho.dat
```

とする．なお，使用する辞書ファイルは特殊文字がエスケイプ2文字方式ではなく未使用記号方式のものでないと正確なデータは出せない．

さて，この節ではこの jisho.dat を実際に利用することを考えてみたい．一般の辞書の見出し語をもとに作成しているので，内容的に偏りのない語彙のはずで，「ドイツ語の語彙」の標本として利用できることが期待できる．一般的な利用法は，やはり，検索が主体であろう．テキストファイル版なら正規表現をつかった検索が可能で，これについて[4.1]で述べる．また，awk には検索機能

だけでなく，集計機能もあり，これがテキストファイル版の語彙リストを利用する上でとても役に立ちそうである．awk の集計機能については[4.2]で述べる．

4．1 正規表現をつかった検索

語彙中に含まれる特定の文字や形態素の調査をしたいならテキストファイルの語彙リストはとても役に立つ．具体例をいくつかあげよう．弱変化動詞で語幹が t や d で終わるものは，文法の学習ではくり返し登場する．現在人称変化で st や t のかわりに est や et の人称語尾をとるし，過去分詞のつくり方でも語末に t のかわりに et をとる．また，命令形ではそれほど確かな規則性はないようだが，やはり，語幹に e のついた形が使われたりする点が他の動詞とことなる．授業でこういう動詞の練習を学生とやっていて例をさらにあげたくなることがある．ところが，これが意外に難しい．-ten や-den で終わる弱変化動詞と言われても arbeiten や baden や beten や heiraten ぐらいはすぐに思い付くが，ほかにはなかなか思いつかない．こういう場合に役に立つのが逆引き辞典であるが，grep とテキストデータなら，検索はもっと容易で迅速にできる．

```
^[a-z].*[dt]en$
```

妙な記号を使った書き方をしているが（ウムラウトなどの特殊文字のことはここでは考慮に入れていない），これは正規表現という書き方になっていて，「アルファベットの小文字で始まる行で den または ten で終わる行」という意味である．正規表現とは，なぜこんな命名になるのか理解に苦しむが，「行の先頭」だとか「行末」だとか，「a から z までの任意の 1 文字」だとか「前の文字の 0 回以上の繰り返し」といった通常の文字情報以外の情報を特殊記号であらわした表記法のことである．正規表現を使えば正確な検索が迅速にできる．たとえば，行末の en と行頭の en や行中の en が正規表現では区別できて，「en$」と書けば englisch の en や denken の最初の en は検索対象から除外される．grep ではこういう正規表現が使えるものと使えないものがあるが，使えるものでは効率的に一行一語の語彙リストから-den あるいは-ten で終わる単語を探してくれる．もちろん動詞以外も検索の網にひっかかって，形容詞の irden や verschieden や副詞の selten や hinten なども検索されてしまうが，これはやむを得ない．

ふつうの検索では語頭からあるいは語末から検索できれば十分だから従来の

紙の辞書や紙の逆引き辞典でも同じように検索できる．テキストファイルの語彙リストとパソコン検索の利点はそのスピードと検索結果がさらにパソコン上で処理可能という便利さぐらいである．ただし，テキストファイル版の語彙リストならではという利用法もある．頻度などの集計については次節で述べるが，話を検索に限っても，たとえばförm を含む語をしらべるというのは機械可読のテキストファイル版ならではの利用法と言えよう．Erk（1985）の語彙家族（Wortfamilie）のインデクスをしらべると，förm に続く要素は約 25 万 4 千語のコーパスで ig が続くものしか見つかっていない．förm を含む異なり語は合計 20 語で延べ 40 語が 25 万 4 千語にあったわけだが，すべて ig が後続している．それでは，förm のあとに ig 以外がくる単語は存在しないのだろうか．こういうことをしらべようと思ったら，紙の辞書は頼りにならない．なぜなら，語頭が förm となっているとは限らないし，語末にしても ig 以外があるかどうかしらべたいのだから辞書の引きようがないわけだ．テキストファイル版の辞書ならこんな検索作業はなんでもない．エスケイプ 2 文字方式の辞書ファイルなら正規表現で「[Ff]¥¥orm..*」を使うだけである．実は「¥」は正規表現としては後続の特殊文字の意味を打ち消す働きのエスケイプ文字なので，ウムラウトの「¥」がこの正規表現の「¥」でないということをあらわすために前者を後者で打ち消して「¥¥」という書き方になっている．jisho.dat でこれをしらべると，やはり，56 語中 52 語では ig が後続していた．förm もしくは Förm で ig が後続してないのは，förmlich, unförmlich と Förmlichkeit と Förmlichkeiten の 4 語だけだった．最後の 2 語については [3.3] で書いたように，元にした 3 冊の辞書の中にどうやら複数形の見出し語を採用していたものがあったようである．整理中のもっと大きな辞書ファイルでしらべてみても ig 以外が後続するものはほとんどふえないようだ．あとは lanzettförmlich と Sandförmchen ぐらいである．

また，awk を使うならさらに特殊な検索も可能で，たとえば，語彙レヴェルの回文が簡単にしらべられる．回文というのは必ずしも文の一種ではなく，「逆さ（さかさ）」や「田植え歌（たうえうた）」や「新聞紙（しんぶんし）」のようなどちらから読んでも同じ語や文などをさすものらしい．ドイツ語では回文に相当する概念に Palindrom がある．厳密に言うと，Palindrom というのは Neger, Lager, Nebel, Beil, Eva, Amor, Koma, Dreh のように逆から読むと別の語になるものも指すようだ．狭義の語彙レヴェルの Wort-Palindrom は逆から読んでも同一語で Anna，Otto, Madam, Reliefpfeiler などが知られている．

awk を使えば狭義の Wort-Palindrom をしらべることは容易だ．[2] にある rev.awk というスクリプトを少しいじればいい．要するに各行の単語を反転させ元の語と同一であれば出力するというスクリプトを書けばいいわけだ．ただし，Anna と annA では同じにならないから単語を小文字に変換して比較している．

```
# KAIBUN.AWK
  {rev=""
   komoji=tolower($0)
   for (i=length($0) ; i>0 ; i--)
     rev=rev sprintf("%s", substr(komoji, i, 1))
   if (komoji==rev)
     print $0
  }
```

結果を見ると，4 文字以上のものはやはりあまり見つからない．あまりに特殊だと思われる語を除くと，Ebbe, Kajak, Renner, Rentner, Retter, Rotor, neben, nennen, neppen, stets などが出てきた．

4．2 awk による集計

awk は grep のように文字や文字列の検索にも使えるし，スクリプトを用意すれば複数の検索が一括しておこなえる．使用した正規表現を出力するようにしておけば，後から修正したり，点検したりするにも便利である．さらに，awk には grep や sed にはない計算機能があって，各種の集計に使える．検索と集計を組み合わせることも可能だが，ここでは，awk による集計の例として語彙の長さ，つまり文字数についての統計と文字頻度についての統計をとっておこう．16 文字のところで結果を折り畳んでいる以外は，awk の出力をそのまま使用している．

1文字：	2語 (0.004%)	16文字：	1238語 (2.232%)
2文字：	47語 (0.085%)	17文字：	849語 (1.531%)
3文字：	369語 (0.665%)	18文字：	540語 (0.974%)
4文字：	1297語 (2.338%)	19文字：	375語 (0.676%)
5文字：	2485語 (4.480%)	20文字：	272語 (0.490%)
6文字：	3970語 (7.158%)	21文字：	160語 (0.288%)
7文字：	4549語 (8.202%)	22文字：	88語 (0.159%)
8文字：	6008語 (10.832%)	23文字：	43語 (0.078%)
9文字：	7259語 (13.088%)	24文字：	36語 (0.065%)
10文字：	7110語 (12.819%)	25文字：	21語 (0.038%)
11文字：	6123語 (11.040%)	26文字：	16語 (0.029%)
12文字：	755語 (8.573%)	27文字：	9語 (0.016%)
13文字：	3496語 (6.303%)	28文字：	5語 (0.009%)
14文字：	2483語 (4.477%)	30文字：	2語 (0.004%)
15文字：	857語 (3.348%)		

これだけの作業をしてくれるスクリプトだが，次にあげるように意外に簡単である．

```
# MOJICHOU.AWK
        {arr[length()]++
        }
END  {
  for (i in arr){
        aver=arr[i]*100/NR
        printf "%2d%-5s%6d%s(%6.3f%%)¥n",i," 文字:",arr[i],"語",aver
        }
  }
```

mojichou.awk を awk で実行して，実行結果を mojichou.dat に保存するには，

awk -f mojichou.awk jisho.dat | sort -n > mojichou.dat

とする．「| sort -n」の部分は使用する awk の種類によっては不要だが，出力の各行を数字の順番で[5]並べかえる意味がある．

語彙の文字数について上の結果を見ておくと，55464語のデータで最頻値は9文字で9文字の語彙を中心にすそ野が両側に伸びている．平均値はこの節の冒頭に書いたように10文字ちょっとになる．最大値が30文字でこちらのすそ野のほうが最小値1文字に伸びるすそ野よりもやや長くなっている．9文字の語彙は最大派閥であるが，これは辞書の見出し語としての話で，通常のテキストで文字数をしらべると，一番頻度が高くなるのが3文字の語彙で，3文字より文字数が大きくなればなるほどそれにつれて頻度は小さくなる．100万字強17万語弱の実用文例集（詳細は後述）でしらべた結果では，30文字の語彙（10語で0.006%）までしらべて9文字の語彙が8文字の語彙よりも頻度がやや高くなる他は文字数が増大するにつれ頻度が例外なく減少している．英語の結果が長尾(1983, p. 25)にのっているが，一番頻度が高いのはやはり3文字の語彙で，文字数がふえるにつれて頻度が小さくなる点も一致している．

語彙の文字頻度もawkで出しておこう．語彙全体の文字頻度と語頭の文字頻度と語末も文字頻度の3つをしらべる．awkのスクリプトはここにはあげないが，3つつくって，結果を汎用ツールのpasteで合成している．なお，awkを使わなくとも多少面倒だが汎用ツールを組み合わせるだけでも文字頻度の計算はできる．全体の文字頻度ならfoldの-1指定でテキストファイル全体を一行一字に折り返すことができるからsortとuniqの-cオプションを組み合わせると文字頻度がかぞえられる．語頭の文字頻度ならcutの-c1指定で1桁めの文字を切り出して，sortとuniqの-cオプションでやはり同じように語頭文字の頻度を計算することができる．cutの-cオプションを-c2とすれば2字めが切り出せるし，-c3なら3字めが切り出せる．語末の文字頻度をかぞえる場合は各語の文字数がバラバラなのでこの方法で語頭からの桁数を指定して文字を切り出すことはできない．revでまず行を反転させて語末の文字を語頭にしてからcut -c1で1字めを切り出せばよい．

	a．文字頻度（全体）	b．文字頻度（語頭）	c．文字頻度（語末）
1．e	79118(14.217%)	s 7126(12.848%)	n 13561(24.450%)
2．n	47520(8.539%)	a 3958(7.136%)	e 7493(13.510%)
3．r	44349(7.969%)	b 3562(6.422%)	t 6922(12.480%)
4．i	37439(6.727%)	k 3289(5.930%)	g 6222(11.218%)
5．t	36773(6.608%)	g 3080(5.553%)	r 5776(10.414%)
6．s	35722(6.419%)	h 2884(5.200%)	h 2886(5.203%)

7. a	33450(6.011%)	v	2823(5.090%)	l	2671(4.816%)
8. l	27824(5.000%)	e	2694(4.857%)	s	1881(3.391%)
9. h	27765(4.989%)	f	2444(4.406%)	d	1726(3.112%)
10. u	22766(4.091%)	w	2417(4.358%)	m	1144(2.063%)
11. g	21108(3.793%)	m	2308(4.161%)	k	1080(1.947%)
12. c	18672(3.355%)	r	2191(3.950%)	z	708(1.277%)
13. o	15093(2.712%)	z	2188(3.945%)	f	670(1.208%)
14. b	13896(2.497%)	p	2125(3.831%)	i	581(1.048%)
15. m	13502(2.426%)	t	2081(3.752%)	ß	475(0.856%)
16. k	13408(2.409%)	d	2040(3.678%)	a	466(0.840%)
17. f	12680(2.278%)	l	1823(3.287%)	o	298(0.537%)
18. d	12662(2.275%)	u	1788(3.224%)	u	255(0.460%)
19. p	9248(1.662%)	n	1434(2.585%)	b	240(0.433%)
20. z	7470(1.342%)	i	917(1.653%)	v	149(0.269%)
21. w	6752(1.213%)	o	631(1.138%)	p	117(0.211%)
22. v	4942(0.888%)	ü	498(0.898%)	x	55(0.099%)
23. ä	4327(0.778%)	j	408(0.736%)	y	49(0.088%)
24. ü	4089(0.735%)	c	339(0.611%)	é	17(0.031%)
25. ß	1787(0.321%)	q	181(0.326%)	c	9(0.016%)
26. ö	1685(0.303%)	ö	99(0.178%)	w	5(0.009%)
27. j	700(0.126%)	ä	90(0.162%)	ö	4(0.007%)
28. y	695(0.125%)	x	26(0.047%)	ü	3(0.005%)
29. x	445(0.080%)	y	18(0.032%)	à	1(0.002%)
30. q	350(0.063%)	à	1(0.002%)		
31. -	242(0.043%)	œ	1(0.002%)		
32. é	24(0.004%)				
33. '	4(0.001%)				
34. œ	3(0.001%)				
35. à	1(0.000%)				

語末文字の頻度表なら Muthmann (1988) の逆引き辞典にもあるが，語彙をひとつひとつかぞえるのでなくページ数をかぞえたらしいこと，長い単語が余計にスペースをとっていたりするので正確に数えることが難しいことなど指摘し

ている(p. 65)．パソコンで処理がおこなえるテキストファイルの語彙リストなら正確にしかも素早く頻度を計量することができる．Muthmann の結果と上位20位の文字の順位を比較しておこう．「城岡1」が jisho.dat のデータだ．

（語末）

	1			5				10				15					20
Muthmann:	n	e	t	g	r	l	h	s	d	m	k	z	f	i	a	ß	o	b	u	v
城岡1：	n	e	t	g	r	h	l	s	d	m	k	z	f	i	ß	a	o	u	b	v

そう違いはないが，h と l の順位，ß と a の順位，u と b の順位が入れ替わっている．

次に語彙全体の文字頻度と語頭の文字頻度について上位20位の文字を出してみよう．jisho.dat のデータは辞書の見出し語データなので使用頻度の高い語彙も低い語彙も一回ずつしか記載されていない．通常のテキストを対象に文字頻度を出せば同一語彙が何度もあらわれるし，変化形もあらわれ，またかなり違った結果になるとも予想される．そこで，ここでは通常のテキストの文字頻度の結果も出しておく．「城岡2」として結果はしめしてある．調査したのは Rossipaul Verlag から出ている下記のフロッピー版実用文例集5点だ．

a．Der elektronische Geschäftsbrief.
b．Der elektronische Geschäftsvertrag.
c．Der elektronische Privatbrief.
d．Die elektronische Stellenbewerbung.
e．Der elektronische Beschwerdebrief.

内容が Windows のテキストファイルになっているので，ウムラウトなどの特殊文字を変換するだけで日本語の MS-DOS 環境でもそのまま処理することができる．合計すると，アラビア数字などを除いて100万字をちょっと越えるデータ量になった．また，藤澤（1984）には「ドイツ・冬物語」という文学作品を対象にした文字頻度の計量結果があるのでこの結果も加えておこう．語彙全体の文字頻度と語頭の文字頻度をパソコンと BASIC で集計したものである．さらに，語彙全体の文字頻度なら英語や仏語のデータもあるので，参考に出しておこう．長田(1985)はニューズウィーク10万字をもとにしたデータであるが，表では「英語1」と記載する．それから，長尾（1983, p. 21）にある英語の文字

頻度のデータはRezaの調査によるとされたものだが，これを「英語2」とする．前川（1995）にも小説，随筆，評論，童話，科学解説の分野からの約10万字のデータをもとにした英語の結果がある．これは「英語3」としよう．また，前川（1995）にはレイモンド・カーヴァーの小説の文字頻度も出ているので，これを「英小説」とする．最後にフランス語のデータだが，これは長田（1986）にあるものだが，データの出所についてはよく分からない．

（全体）

	1				5					10					15					20
城岡1：	e	n	r	i	t	s	a	l	h	u	g	c	o	b	m	k	f	d	p	z
城岡2：	e	n	r	i	t	s	a	h	d	u	g	l	c	m	b	o	f	w	z	v
藤澤：	e	n	i	r	s	t	h	d	a	l	c	u	m	g	o	b	w	f	k	z
英語1：	e	t	a	o	i	n	r	s	h	d	c	l	m	p	u	f	g	w	y	b
英語2：	e	t	a	o	i	n	s	r	h	l	d	u	c	f	m	w	y	p	g	b
英語3：	e	t	a	o	i	n	h	s	r	d	l	u	w	y	g	m	c	f	b	p
英小説：	e	t	a	o	i	n	h	s	r	d	l	u	m	w	k	y	g	c	b	v
仏語：	e	a	i	s	t	n	r	u	l	o	d	m	p	c	v	q	b	f	g	j

（語頭）

	1				5					10					15					20
城岡1：	s	a	b	k	g	h	v	e	f	w	m	r	z	p	t	d	l	u	n	i
城岡2：	d	a	s	i	e	v	b	w	m	g	u	f	z	h	n	k	p	l	r	t
藤澤：	d	s	i	w	m	e	g	a	h	u	b	v	k	n	f	l	z	r	t	j

全体の文字頻度では「城岡1」と「城岡2」ではかなり一致している．つまり，辞書の見出し語データと通常のテキストの文字頻度は上位で見るかぎりかなり一致する（この場合は7位まで完全に一致している）もののようである．藤澤の結果を見ても，文学作品を対象にしたものであるが，全体の文字頻度では「城岡1」や「城岡2」との類似は，英語や仏語のデータを見比べるまでもなく否定しがたい．1位と2位の文字は一致しているし，上位6位までの文字のメンバーの顔ぶれは同一だ．一方の語頭の文字頻度は，辞書の見出し語データと通常のテキストではずいぶんことなっているようだ．まず，dの頻度で鋭く対立する．「城岡1」で16位のdが「城岡2」では1位だ．当然これは定冠詞のder,

die, den, des, das, dem や接続詞 daß の高頻度に由来している．同様に i の頻度も「城岡 1 」で 20 位が「城岡 2 」で 4 位といちじるしく順位をあげている．in, ich, ist, im などがこの順位のアップに貢献しているようだ．逆に k では 4 位から 16 位，h では 6 位から 14 位，r では 12 位から 19 位と「城岡 2 」でつまり通常のテキストで大幅に順位を下げている．「城岡 1 」と「藤澤」をくらべてみても，d と i がいちじるしく順位を上げ，k, h, r で順位を落とすという図式は変わらない．一方，通常のテキスト同士の比較では，「城岡 2 」と「藤澤」が一見してかなり異なる文字頻度の分布をしている点も見過ごせない．このあたりは今後の研究を待ちたいが，語頭の文字頻度については文体的な差異が出る可能性もありそうだ．

5．最後に

現在，様々な規模のドイツ語のテキストファイル版逆引き辞典を編集中だ．紙の逆引き辞典並の大規模な逆引き辞典が完成すれば，需要があるようなら公開して，他の研究者に自由に利用してもらうつもりでいる．また，本稿で述べた方法はドイツ語だけでなく，適当な電子ブックさえ手に入れば他の外国語にも応用できる．私自身は，日本語や英語のものも作成していて，そのうちにドイツ語と日本語や英語との比較研究をおこなうつもりでいる．研究が進みしだい成果を発表していきたい．

注

1）手間から言えばこちらのほうが楽なのでこの方法についても述べておこう．全文検索では何も指定せずに検索させると辞書の先頭から検索を始めて 32,000 項目まで検索してくれる．検索終了後に見出し語を選択状態にしてからメニューバーの「編集」で「エディター起動」を選択する．次に出てくる表示で出力したいファイル名を指定して，「該当項目見出しのみ」をチェックする．そして，「OK ボタン」を押せばテキストファイルに見出しを出力してくれる．ただし，DDwin では一度に出力できるのは 32,000 項目という制限があるため一度ではすべての見出し語が取り出せない場合もある．その場合は，検索の開始位置を指定して上記の作業を繰りかえすことになる．この検索の開始位置の

指定は見出し語を取り出すためにはなかなか便利で，たとえば英和辞典と和英辞典がいっしょになっている電子ブックで検索開始位置を「あ」に設定すれば日本語の見出し語だけが取り出せる．ただし，ここで述べた方法が使えない電子ブックもあるようだ．

2）Universalwörterbuch の A の全項目と B では Bahnwärter までがテキストファイルとしてボン大学にあり，D. Hein (1995) ではそれを利用しているのだが，DDwin を利用すれば辞書全体がテキストファイルとして研究に利用できる状況になっている．

3）なお，grep や awk を活用すれば，わざわざ配列を変えたファイルをつくらなくともこういう調査はできるのだが，実際に目で確認したり，同時に他の色彩語についてもしらべたいというようなときにはここで述べた方法が役に立つ．

4）sed は削除や変換に使われることが多いと思われるが，grep や awk のように検索に利用することもできる．しかも，sed では awk では不可能なタグ付き正規表現がつかえるので畳語の検索なども簡単に実現できる．すべてを小文字になおしたファイルを jisho.dat とすれば，plemplem のような 4 字以上の文字列が繰り返される畳語をしらべるには，

```
sed -n "/^¥(.....*¥)¥1$/p" jisho.dat
```

のようにする．こういう作業は語彙数の大きいリストを対象にしないと結果はえられない．クラウン独和辞典では plemplem でさえ見出し語にはなっていない．整理中のデータファイルのうち最大の 17 万語程度のものを利用した結果を示しておこう．大文字で始まる語は大文字に戻して示すと，plemplem 以外では，ballaballa, Matamata, Pinkepinke, killekille, Töfftöff, Beriberi, Dividivi, tucktuck, hopphopp, Couscous, Froufrou, Kompetenzkompetenz が見つかった．日本語とは比べものにならないほど少ないようだ．

5）「数字の順番で」というのは分かりにくい言い方だが，数字の並びを先頭の文字から比較すると，たとえば「3」よりも「12」のほうが小さい．文字列としてソートすると事実「12」のほうが「3」よりも小さいものとして並べかえられてしまう．それゆえ，sort では数字として並べかえたい場合には-n オプションをつけることになっている．

6）データの全文字数を計算するのに wc というツールが使えるが，ふつうの wc では改行文字も数えるので一行一語の語彙リストを使用して文字数をかぞ

える場合には無視できない誤差が生じてしまうので注意が必要だ．実用文例集の100万字程度のデータで一行一語の語彙リストをつくると，17万行ぐらいになったので，wcの種類によっては文字数で17万字程度は文字数が増えてしまう．高取和民さんがMS-DOS用に作成したwcなら改行文字や空白文字は無視してくれるようだ．

参考文献

石井正彦（1992）：プログラムを書かずにできる言語処理「日本語学」1992年9月号，明治書院，pp. 114-124

石川克知（1994）：コンピュータによるテキスト処理「言語文化紀要」27号，北海道大学，pp. 77-100

岩波書店辞書編集部編（1992）：「逆引き広辞苑」岩波書店

植田康成（1982）：計算機によるドイツ語教科書の語彙調査について「広島大学文学部紀要」42巻特輯号3号，pp. 1-162

植村富士夫・富永浩之（1993）：「awkでプログラミング」オーム社

片山裕（1993）：「sedパズルブック」インプレス

河野収（1986）：インデックス・コンコーダンスの機械処理「同志社外国文学研究」43・44合併号，同志社大学外国文学会，pp. 1-17

菊池雅子（1983）：初級ドイツ語教科書の語彙「日吉論文集」32号，慶應義塾大学商学部，pp. 86-113

北原保雄（1990）：「日本語逆引き辞典」大修館書店

金水敏（1992）：AWKによるテキスト型データベース活用術「日本語学」1992年6月号，明治書院，pp. 127-135

佐々木巧昌（1991）：データディスクマン「日本語学」1991年8月号，pp. 91-94

杉本武（1992）：正規表現によるプレーン・テキストの検索「日本語学」1992年7月号，pp. 112-121

SE編集部（1992）：「MS-DOSテキストデータ料理学」翔泳社

田島毓堂・丹羽一彌（1987）：「日本語尾音索引」笠間書院

長尾真（1983）：「言語工学」昭晃堂

長田順行（1985）：「暗号」現代教養文庫

長田順行（1986）：「暗号と推理小説」現代教養文庫

羽山博（1991）：「実用 UNIX」アスキー
藤澤正明（1984）：パーソナルコンピュータによるドイツ語処理「研究報告（人文・社会科学）」32号，九州工業大学，pp. 83-98
舟本奨（1994）：「実用 UNIX ハンドブック［改訂新版］」ナツメ社
古田啓（1991）：文法研究のためのデータベース利用「日本語学」1991年12月号
前川守（1995）：「1000万人のコンピュータ科学3　文学編　文章を科学する」岩波書店
室井禎之(1992)：コンピュータ利用によるコロケーション分析の実際とその言語研究上の意義について「言語文化部紀要」22号，北海道大学，pp. 153-167
森泉（1991）：パーソナルコンピュータによる初級独作文教材の語彙調査「慶應義塾大学日吉紀要ドイツ語学・文学」12号，pp. 105-121
米井巌(1986)：パーソナルコンピュータを用いたドイツ語初級教材の語彙調査「研究紀要」32号，日本大学人文科学研究所，pp. 160-193
米井巌（1989）：逆引き形式の語彙総覧とその語学教育上の有効性「ドイツ文学論集」10号，日本大学，pp. 41-59

Aho, A. V./Kernighan, B. W./Weinberger, P. J. (1989)：「プログラミング言語 AWK」トッパン
Braun, P. (1993): Tendenzen in der deutschen Gegenwartssprache. 3. Auflage. Stuttgart (Kohlhammer).
Dougherty, D. (1991)：「sed & awk プログラミング」アスキー
Erk, H. (1985): Wortfamilien in wissenschaftlichen Texten. München (Hueber).
Hein, D. (1995): UNIX gestützte maschinelle morphologische Untersuchung zur Komposition. Bonn (Holos Verlag).
König, W. (1978): dtv-Atlas zur deutschen Sprache. München.
Mater, E. (1965): Rückläufiges Wörterbuch. (VEB Bibliographisches Institut).
Haas, W. [Hrsg.] (1984): Textkorpora 1. PHONAI Band 28. Tübingen (Niemeyer).
Meier, H. (1964): Deutsche Sprachstatistik. Hildesheim (Olms).

Muster, J./Birns, P. (1992) :「UNIX コマンド活用ハンドブック」パーソナルメディア

Muthmann, G. (1988) : Rückläufiges deutsches Wörterbuch. (Niemeyer, RGL-Serie).

Pregel, D./Rickheit, G (1987) : Der Wortschatz im Grundschulalter. Hildesheim (Olms).

Ruoff, Arno (1990) : Häufigkeitswörterbuch gesprochener Sprache. 2. Auflage. Tübingen (Niemeyer).

Stallman, R. M./Rubin, P. H./Robbins, A. D./Close, D. B. (1993) :「GAWK」アジソン・ウェスレイ

Staubach, G. (1989) : UNIX-Werkzeuge zur Textmusterverarbeitung. Berlin Heidelberg (Springer).

Wehrle, H. (1993) : Deutscher Wortschatz. Stuttgart, 16. Nachdruck.