An Experiment in Automatic Word Counting with Homonym Differentiation by the Use of a Computer

電子計算機による語彙調査の一実験

石　綿　敏　雄

1　語彙調査自動化の試み

語彙調査は，語彙の記述の一方法として有効なものであると考えられるが，同時に，いろいろな応用面ももっており，言語調査のなかでも重要なもののひとつであるということができる。ところで一般に語彙調査は多くの労力を必要とするものであり，特に大規模のものは容易に行ないがたいものである。そこでこのために電子計算機を用いるということが考えられる。

語彙調査を電子計算機を用いて行なうにも，かなりさまざまな方法が考えられる。たとえばある言語作品をとりあげたとき，そのすべての単語をひとつひとつとりあげて見出しとし，それぞれにその属する文脈や出典所在などを付記して一件の記録単位にし，見出しについて字母順の排列を行ない，はじめからその文脈を印字してゆく，という方法もある。このようにすれば用例つきの総索引といったようなものができるわけである。このような方法は大変簡単でよいが，機械操作のあとで同じ見出しの中での同音語をふるいわけたり，ばらばらになっている活用語をひとところに集めたりする，というような作業を人間がしなければならない。また単位が「電信｜ばしら」のように切れていたばあい，「ばしら」は「はしら」に合わせておいたほうがよいであろう。このような作業を，用例すなわち付けられた文脈をもとにして，それを読んで人間が判断する，ということをしなければならない。このような方法は計算機にかけるばあいのプログラム作成は容易であるが，人間の労力がかなりかかる。

次に，このような操作まで含めて電子計算機にさせようという方式も考えられる。同音語を見分けたり，活用したいくつかの語形を同一のものにまとめたり，いくつかの語形をひとつにまとめたりするというようなことは，語彙調査の一過程である。このようなことを計算機で行なうという方法も考えられよう。これが

機械化できれば人間の手間をある程度まで省くことができる。またもう一方には，自動抄録，機械翻訳というような言語情報処理を行なうばあいでも以上のような操作は当然行なわれなければならないのであるから，その最初の段階として，そのような方面の開発にも寄与することができるであろう。ことに，自動抄録のアイデアのひとつとして語彙の分布を手がかりにしようとするものがあるが，そのようなばあいには語彙調査をすることによって語彙の分布を知るわけであり，その意味で自動抄録プログラムの全過程の一部として語彙調査プログラムが含まれる形になるので，語彙調査が自動化されるということは自動抄録プログラムを作る上で重要な意味があるといえよう。語彙調査の自動化には上に述べたような種種の点で重要性がみとめられる。

そこでこの後者のような語彙調査の自動化ということをめざしたものについて考えてみたいと思った。このようなばあいには，さまざまな方法を考えてみて，これをプログラムに組んで，少量のデータを用いて実験してみる，というような試みを繰り返すことが必要であろう。そこでそのような，ひとつの実験を行なってみた。以下これについて述べる。

2　使用したデータと設定した問題

データとして次の文章を取りあげてみた。

YOOKO WA NIZYUU SITI SAI NI NARU .

KYOOTO NO DAIGAKU NO EIBUN' KA WO DE TE NIZYUU SAN' SAI NO TOSI NI NIQPON' KIN'Z/ OKU KOOGYOO NO HISYO KA NI NYUUSYA SI TA .

DOOKI NO TOMODATI GA MASUKOMI KAN'KEI WO SIBOO SI TA NO NI TAISI TE YOOKO GA ZIM/I NA KAISYA ZUTOME WO ERAN' DA NO WA SONO YOONA SEIKAKU KARA DE ARU SI SORE NARI/ NO ZIZYOO MO AQ TA .

YOOKO WA NI DO HODO NIQPON' KIN'ZOKU KOOGYOO NO ZIGYOO HON'BU TYOO NI AQ TA KOTO/ GA AQ TA .

単位分割も重要な課題であるが，別に考えることにして，ここではあらかじめ言語単位の区切りを入れておいた。（文中のスペースは語の切れ目，/ はカードの切れ目である。）この例文はローマ字により IBM80 欄カードにさん孔したが，

これはこのプログラムを組み始めたときの機械的な条件によるにすぎない。データ入力がテープになってもプログラムのほんの一部を変えるだけですむ。（かなを用いるばあいには辞書やサブルーチンの一部を変えるだけで，漢字を用いるばあいにもそれに加えて命令の一部を変更するだけで，大体はこの実験の行き方で実行可能であると考えられる。）

さて上のデータを選んだのは，語彙調査のばあいにおこる次のような問題を含んでいると考えたからである。

(1) 同音語があること。

AQ　「……にあったことがあった」「事情もあった」――「会う」と「ある」

DE　「大学を出て」「事情からであるし」――動詞「でる」の連用形と助詞「で」

KA　「英文科」と「秘書課」

NI　「二十七才になる」「志望したのに対して」と「二度ほど」――助詞「に」と数詞「二」

SI　「入社した」「志望した」と「性格からであるし」――動詞「する」の連用形と助詞の「し」

(2) 活用形がばらばらになっていること。

AQ　「それなりの事情もあった」「会ったことがあった」「性格からであるし」のように連用形2例，終止形1例と存在するが，これをまとめると3例あったことになる。

このほかに「会ったこと」を「会う」，「入社し」「志願し」を「する」，「に対して」を「対する」，「選んだ」を「選ぶ」，「ような」を「ようだ」に直すような操作もしておく。つまり終止形でまとめてみる。

(3) 活用語以外でも語形の整理上必要なものがあること。

たとえば「会社づとめ」の「づとめ」は「つとめ」に直す（ZUTOME→TUTOME）。

以上に述べたような問題があるが，これらを人間が行なうのと同じように計算機に行なわせる，つまり，同音語を見分けたり，活用語やちがった語形をまとめたりするというような言語学的な操作まで電子計算機に行なわせようというわけである。

3　辞書とプログラムの全体的な設計

このような実験を行なうためには，機械にことばについての知識を与え，同音

語の判別法その他を教えなければならない。ここでは辞書を作ることと各同音語の判別のための特別な小プログラム（サブルーチン）を作るという方法によってこれを解決しようと考えた。

辞書には次のような事項を書き入れることにした。

見出し・同音語の有無。品詞。文法的な各動の情報。語彙調査の見出し。

はじめの見出しはテキストの中に出てくる形(単位語)であり，後ろのは語彙調査上立てるべき見出し(見出し語)である。(「単位語」「見出し語」については，国語研究所報告13「総合雑誌の用語」後編95ページのそれに大体近いと考えてよい。)

実験に使用した計算機は，日本ビジネスコンサルタントの HITAC 3010（主記憶装置コア20000けた）である。（辞書に記入するデータの長さはすべて variable length)。その記入例を一，二示す。

AIDA(0*－*N*C↑－)
AQ(0*#001*V*2↑ARU&V*2↑AU)
ARU(0*－*V*3↑－)
AU(0*－*V*3↑－)
⋮
KYOOTO(0*－*N*PROPER↑－)
MAINITI(0*－*N*C↑－)
WO(0*－*PAR*CASE↑－)
ZUTOME(0*－*N*C↑TUTOME)

全部で百語ほどのものを作成した。WO を例にして説明すると次のようである。WO の次の(　)の中にことばの 情報がはいっている。(　) の中の＊は情報の仕切りである。はじめのゼロは無意味（プログラム作成当初ある情報を置こうと思ったが，途中で不要になったので 0 が置いてある)，次の－は同音語なし(同音語のあるものは＃がある。AQ の例をみよ)，次の PAR は品詞でこれは助詞(Particle の略)，次の CASE は格，↑の次の－は単位語の形と見出し語の形が同じであることを示す（もし違えばその形を書く。たとえば ZUTOME の例をみよ)。

文法情報としてはもっと簡潔な示し方もあるしその方がよいと思われるが（たとえば PAR は P で表わす)，この実験では少し長めに使って人間に読みやすくした。

同音語の判別法は文脈によるという方法を用いた。つまり，人間ならば前後の文脈から同音語を判別しえているわけであるが，計算機にもその語の前後を読ま

せて判別させようというのである。計算機であるから実際には調べてということになる。文中での，当面の語の前後の語をとらえて，その単語の文法的な性質を辞書に書いてある情報を利用して調べ，それによって判定を行なった。同音語判別のためのルーチンはサブルーチンの形にして，同音語判別の必要が生じたときこれを用いるという形にした。同音語判別サブルーチンは今度の実験では７個作成したが，本当の調査をするばあいにはもちろんもっと多量にふやさなければならない。このばあいに，そのようなルーチンをどこへしまうかが重要であるが，この実験では磁気テープにしまって必要に応じて呼び出すという方式をとった（後述）。

同音語の判別ルーチンを作るためには，あらかじめありうべき文脈の型を研究しておかなければならない。この実験ではこの例文中のいろんな条件を考えて作成してみたが，文脈の型は非常に種々様々な形で現われうるものであるから，予期しえた文脈のばあいにはよいけれども，予期しえなかったものについては取り扱い次第で結果に誤りが生じかねない，ということがいえる。そこでこのプログラムでは，予期しえない型が現われたら，機械に判断させないでその語をそのままプリントアウトさせ，人間がそこだけは介入して処理する，というようにつくってある。その方が安全だと考えたからである。

判別ルーチンの一例をフローチャート２に示した。これはこれ自身もっと拡大すべきであり，それが可能である（たとえば助詞「に」の前の名詞の種類を問うなど）が，この実験ではこの程度にとどめておいた。

機械の各部分には次のような任務を与えた。

中央演算処理装置――語彙調査主プログラム（メイン・ルーチン）の格納とそれによるデータ処理。サブルーチンの一時的格納。

磁気テープ１――辞書格納。

〃　２――同音語判別ルーチン格納。

〃　３――処理ずみデータ格納。

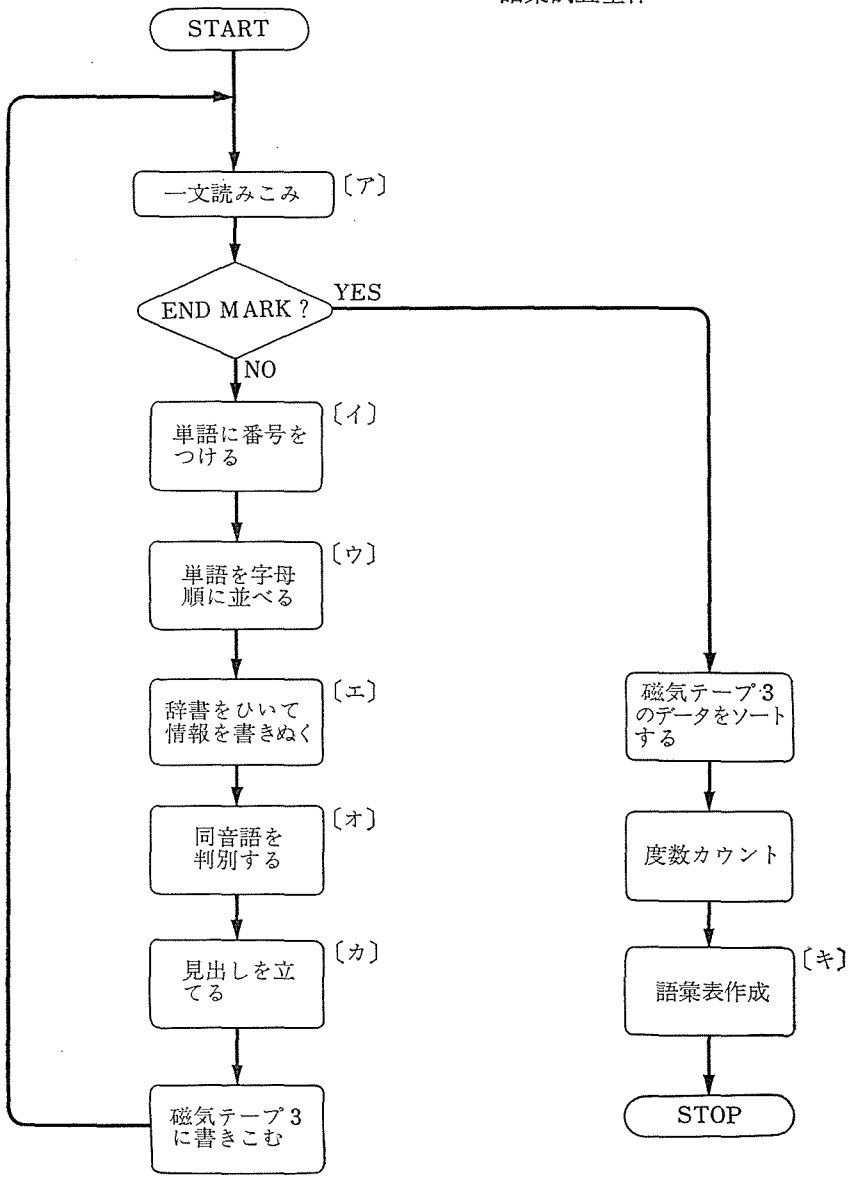
カード読み取り装置――データ(原文)入力。

高速度印字装置――結果としての語彙表作成。

メイン・ルーチンのフローチャートはフローチャート１に示す。これによってデータがどのように扱われるかを，全データについて示すとかなりの紙幅を要するので，ここではデータのなかから一文をとりあげて示そうと思う。このプログラムによってデータがどのように取り扱われるかを説明するために，この一文を

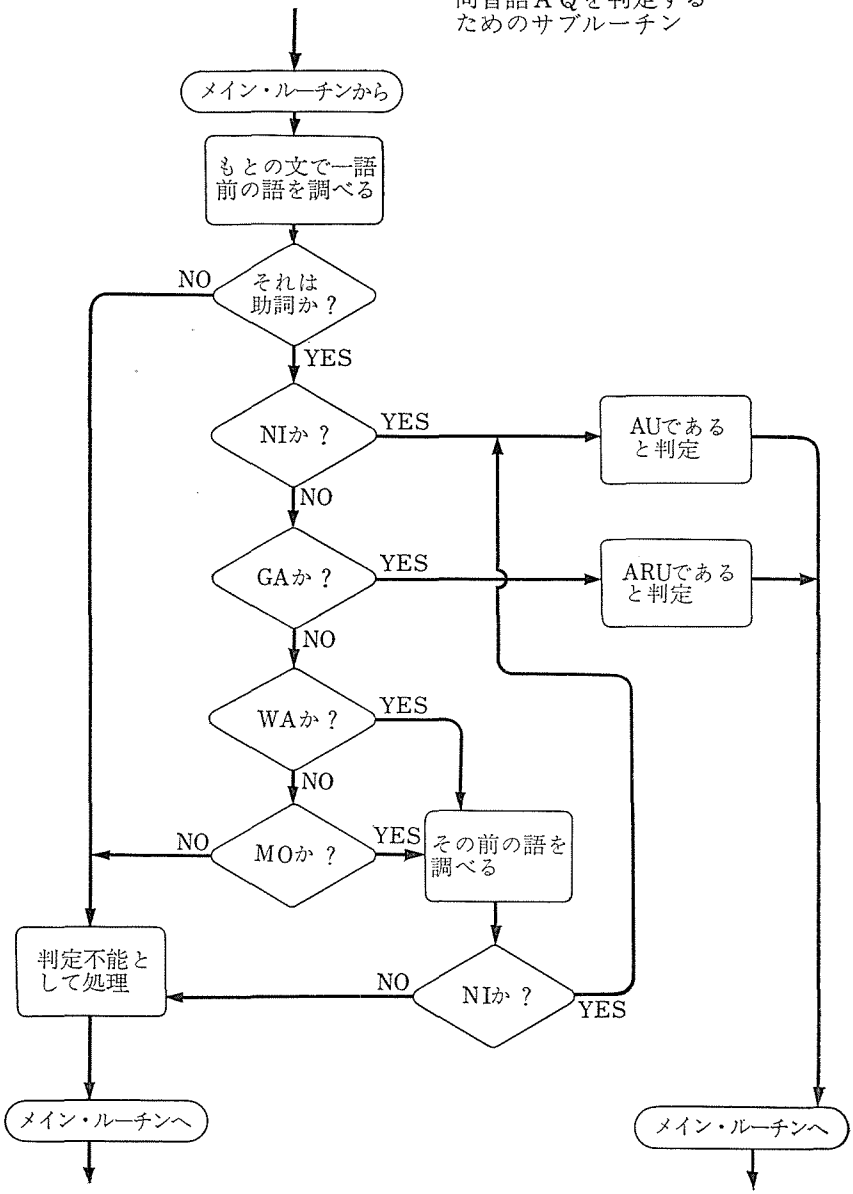
フローチャート　1

語彙調査全体



フローチャート　2

同音語ＡＱを判定するためのサブルーチン



読みこませてデータ処理したばあいの各段階の最終の様子を，計算機の主記憶装置の内容をタイプアウトしたような形で，次に示す。

〔ア〕 カードからカード読みとり装置を通じて主記憶装置 8000 番地にデータを読みこむ。8000 番地以下の内容

KYOOTO NO DAIGAKU NO EIBUN' KA WO DE TE NIZYUU SAN' SAI NO TOSI NI NIQPON' KIN'ZOKU KOOGYOO NO HISYO KA NI NYUUSYA SI TA .

〔イ〕 文中の語のそれぞれに始めから番号をつけて A000 番地以下に転送する。A000 番地以下の内容

KYOOTO/0001(NO/0002(DAIGAKU/0003(NO/0004(EIBUN'/0005(KA/0006(WO/0007(DE/0008(TE/0009(NIZYUU/0010(SAN'/0011(SAI/0012(NO/0013(TOSI/0014(NI/0015(NIQPON'/0016(KIN'ZOKU/0017(KOOGYOO/0018(NO/0019(HISYO/0020(KA/0021(NI/0022(NYUUSYA/0023(SI/0024(TA/0025(.

〔ウ〕 A000 番地以下から・のあるまでのデータを，単語をキー・ワードにしてソートする(ABC 順排列)。キー・ワードの長さは variable。結果は A000 番地以下に

DAIGAKU/0003(DE/0008(EIBUN'/0005(HISYO/0020(KA/0006(KA/0021(KIN'ZOKU/0017(KOOGYOO/0018(KYOOTO/0001(NI/0015(NI/0022(NIQPON'/0016(NIZYUU/0010(NO/0002(NO/0004(NO/0013(NO/0019(NYUUSYA/0023(SAI/0012(SAN'/0011(SI/0024(TA/0025(TE/0009 TOSI/0004(WO/0007(.

〔エ〕 磁気テープ１を動かしてこの文中の単語について辞書を引き，辞書の説明（情報）を書きぬいて 8000 番地以下に並べる。8000 番地以下の内容

DAIGAKU/0003(0*－*N*C↑－)DE/0008(0*#002*PAR*0↑－&V*2↑DERU) EIBUN'/0005(0*－*N*C↑－)HISYO/0020(0*－*N*C↑－) KA/0006 (0*#003*N*C↑－&N*C↑－)KA/0021 (0*#003*N*C↑－&N*C↑－) KIN'ZOKU/0017(0*－*N*C↑－)KOOGYOO/0018(0*－*N*C↑－)KYOOTO/0001 (0*－*N*PROPER↑－)NI/0015(0*#006*PAR*CASE↑－&N*NUM↑－) NI/0022 (0*#006*PAR*CASE↑－&N*NUM↑－)NIQPON'/0016 (0*－*N*PROPER↑－) NIZYUU/0010 (0*－*N*NUM↑－)NO/0002(0*－*PAR*CASE↑－) NO/0004(0*－*PAR*CASE↑－)NO/0013(0*－*PAR*CASE↑－)NO/0019(0*－*PAR*CASE↑－)NYUUSYA/0023(0*－*N*C↑－)SAI/0012(0*－*N*AN↑－) SAN'/0011 (0*－*N*NUM↑－)SI/0024(0*#007*PAR*0↑－&V*2↑SURU)TA/0025(0*－*AU*3↑－) TE/0009

(0*－*PAR*0↑－)TOSI/0014(0*－*N*C↑－)WO/0007(0*－*PAR*CASE↑－).

〔オ〕 同音語を見わけてデータを A000 番地以下へ転送。このばあい，「英文科を出て」の DE と，「英文科」「秘書科」の KA と，「二十三才の年に」「秘書課に」の NI，「入社した」の SI が，同音語があるので判判すべきだと考えたことばである。それぞれ判別ルーチンの 002, 006, 007 を用いて判別するのである。判別した結果は以下に見られる通りで， DERU, 英文科の KA，秘書課の KA，助詞の NI（2例），SURU と判定している。A000 番地以下の内容は

DAIGAKU(0*－*N*C↑－)DERU(0*－*V*3↑－)EIBUN'(0*－*N*C↑－) HISYO (0*－*N*C↑－)KA(0*－*N*EIBUN'－↑－)KA(0*－*N*HISYO－↑－)KIN'ZO KU(0*－*N*C↑－)KOOGYOO(0*－*N*C↑－)KYOOTO(0*－*N*PAR*PROPE R↑－)NI(0*－*PAR*CASE↑－)NI(0*－*PAR*CASE↑－)NiQPON'(0*－*N*PR OPER↑－)NIZYUU(0*－*N*NUM↑－)NO(0*－*PAR*CASE↑－)NO(0*－*PA R*CASE↑－)NO(0*－*PAR*CASE↑－)NYUUSYA(0*－*N*C↑－)SAI(0*－*N *AN↑－)SAN'(0*－*N*NUN↑－)SURU(0*－*V*3↑－)TA(0*－*AUX*3↑－) TE(0*－*PAR*O↑－)TOSI(0*－*N*C↑－)WO(0*－*PAR*CASE*↑－).

〔カ〕 A000 番地以下にあったデータについて見出し語として立てるべきものを立てて 8000 番地以下に転送。このデータについては前の段階まででですんでしまっているので，ここで特に変更はなかった。「会社づとめ」の ZUTOME→TUTOME のような処理がここでなされるわけである。次のソートにそなえて語と語の間にスペースを入れておく。8000 番地以下の内容のはじめの方だけを示す。

DAIGAKU(0*－*N*C↑－) DERU(0*－*V*3↑－) EIBUN'(0*－*……

上の〔カ〕までの処理がすんだら，処理ずみのデータとして磁気テープ3に書きこみ，次のデータの処理すなわち〔ア〕の段階へもどるのである。このようにして次々と文を処理してゆき，データが全部終わりになったら，磁気テープ3から処理ずみのデータを全部出し，全部についてソートする。このようにすると同じ単語がひとところに集まる。これを順次に同じ単語のあるだけ数えてゆけば語彙表ができる。たとえばもし前に例として示したデータだけでカウントするとすれば，これをはじめから見てゆくと，はじめからずっと1語ずつであるが，NI がすべて助詞で2回，NO がすべて助詞で3回あるので次のような語彙表を作る。

〔キ〕 DAIGAKU(0*－*N*C↑－)　1
DERU(0*－*V*3↑－)　1
EIBUN'(0*－*N*C↑－)　1

HISYO(0*—*N*C↑—)	1
KA(0*—*N*EIBUN'—↑—)	1
KA(0*—*N*HISYO—↑—)	1
KIN'ZOKU(0*—*N*C↑—)	1
KOOGYOO(0*—*N*C↑—)	1
KYOOTO(0*—*N*PROPER↑—)	1
NI(0*—*PAR*CASE↑—)	2
NIQPON'(0*—*N*PROPER↑—)	1
NIZYUU(0*—*N*NUM↑—)	1
NO(0*—*PAR*NUM↑—)	3
NYUUSYA(0*—*N*C↑—)	1
SAI(0*—*N*AN↑—)	1
SAN'(0*—*N*NUM↑—)	1
SURU(0*—*V*3↑—)	1
TA(0*—*AUX*3↑—)	1
TE(0*—*PAR*0↑—)	1
TOSI(0*—*N*C↑—)	1
WO(0*—*PAR*CASE↑—)	1

上のような語彙表を高速度印字装置が印字して出すとプログラムが終わりになる。

この論文の「2 使用したデータと設定した問題」で示したデータ全部と他のデータを合わせて210語のテキストについてプログラムを実行してみた。第一文の読みこみから全体についての語彙表を作成し終わるまでに2分23秒を要した。

ここで二つの点が問題になる。その一つは語彙表作成直前のソートに約40秒を要していることである。これは，ソートプログラムを筆者が作成したために，あまり能率のよくないソート法を用いているという事情による。データが非常に多くなったばあいにはこれでは困るわけであるが，そのようなばあいにはその直前でプログラムを一度切断して，ソートの段階だけサービス・ルーチンとして開発されているスピードの早いソート・ルーチンを使えば問題がなくなるわけである。しかしもしデータの量が比較的少ないならば，たとえば論文をひとつずつ自動抄録するというようなばあいならば，データとしてはそう多くないわけであるから，サービス・ルーチンを使わなくてもすむであろう。サービス・ルーチンを使うと，いったんプログラムを切断しなければならないので，全体を自動的に流す

ことができなくなるということもある。この意味でこの実験のプログラムも一応の意義をみとめてよいことと思われる。

ソート・プログラムはこの全ルーチンのなかで二つ使ってあるが，辞書を引く前のソートの方はデータの数が常に少ないわけであり，事実実験のさいにも相当の高速で処理されていた。

さて問題点の第二は，辞書の語数やサブルーチンの数がふえたときにその検索スピードが落ちるわけで，そのようなばあいには上に示した時間よりもずっと多くの時間を要する，ということである。そこでこのプログラムでは辞書の検索についてはひとつの方法を試みているわけであり，そのようなことについて次に述べたい。

4　このプログラムでのいくつかの試み

このプログラムではいくつかの点である種の試みを行なっている。それは次のようなものである。

辞書検索の方法

辞書と同音語判別サブルーチンの格納法

同音語判別ルーチンをサブルーチンとしてメイン・ルーチンから切り離したこと

カウント方式

はじめに辞書引き(テーブル・ルック・アップ)について。テーブル・ルック・アップは通常かなりの時間を要するものである。これの短縮法を考えておくべきであろう。特に辞書は通常かなり大きなものになるので，主記憶装置のなかにはとても収めきれない。補助記憶装置へ入れなければならない。このばあい呼び出し時間が問題になる。いまのところ磁気テープへ収めるというのが一般的であろうから，このプログラムでも磁気テープを利用したばあいについて考えてみたわけである。さて磁気テープによってひとつひとつ単語を検索するためにいちいちリワインドしたりするとそのために相当の間を要する。逆読みができるばあいでも何往復かすることになり，これも時間を要するであろう。そこでこの実験では辞書をあらかじめ字母順に排列しておき，読みこんできたデータについても一度ソートして字母順にならべ，両方をつきあわせるような形でテーブル・ルック・アップを行なわせるという方法をとってみた。このためには主記憶装置のなかで

一度ソートするという手間がかかるわけであるが，(それと，そのために単語に番号をつけておいてもとの文中での位置を示すものを残しておく必要があるが)，一文のなかにふくまれる単語の数は大体のばあいそれほど多くなく，かつ主記憶装置内での操作はきわめて早いものであるから，そのためにそれほどの時間を要することはない。そしてそのような方法をとったとき，磁気テープが一回まわればその文のテーブル・ルック・アップは終わってしまという利点がある。

この実験ではこのような考え方をとって実行してみた。実験の結果は良好であった。

このような，ソートしてつきあわせる，いわばソート・アンド・テーブル・ルック・アップという方式は，辞書を早く引くための一つの方法であると考えている。もちろん，ほかにもっと方法があるに違いないし，よりよい方法があるに違いない。またこれは磁気テープを使うばあいのことで，他の装置を使うときは問題がなくなるか，または別のものになろう。

辞書が主記憶装置にはいりきらないのと同様に，同音語を見分けるサブルーチンもまた主記憶装置にはいりきらないに違いない。このプログラムではこれをサブルーチンにし，多数のサブルーチンを磁気テープにしまっておくことにした。必要に応じて必要なルーチンを主記憶装置に呼び出してこのルーチンを実行することによって同音語の判別を行ない，判別が終わったらメイン・ルーチンへもどるように作った。このようにすればかなり多数のルーチンを収容しうることになり，主記憶装置の容量が小さくても大きな仕事が行なえる，すなわちこのままでも最大数十万ステップの大きなプログラムにまで拡大しうるわけである。

同音語判別をサブルーチンにしたのには，また次のような理由もある。一般にこのようなルーチンは量的にもかなり大量のものが必要であり，その作成には多くの時日を要する。また質的にも多くの問題を含み，改良に改良を重ねるという手段をとらざるを得ない。となると全体を一時に完全に作るというわけにはいかないから，順次に補充し改善してゆくという手段をとらざるを得ないであろう。そこでこのプログラムでは，辞書とサブルーチンに手を加えさえすればこれが可能であるように工夫した。つまりメイン・ルーチンは現在のままでいつまでも使うことができるように作ろうとした。このために判別ルーチンはサブルーチンとして独立させることにしたのである。また辞書やサブルーチンについても，新しい情報を辞書に加えたり，新しい判別ルーチンを作ったりしても，一定の方式に従って行なうならばそれまでにできている判別ルーチンの運用に一向さしつかえ

ないというようにした。このために辞書の情報の境目に＊を用い，↑の前にならいくつ書きこんでもよいようになっている。

このように作ってみたので，辞書と同音語判別ルーチンを補充しさえするならば，このままでもかなり多量のデータでも取り扱うことができるはずである。

同音語がテキストのなかに二つ以上並んで出てきたときはどうなるだろうか。前の単語は後ろの単語によって定まり，後ろの単語は前の単語によって決定されるというようなばあいもあるに違いない。この実験でもそれに似たばあいを設定してみた。すなわちデータ中の最後の文の中の NI AQ というような例である。このようなばあいには辞書の記載事項の排列を考えること，あるいはこれを解決するようなプログラムをもうひとつつくるというような手段が考えられるが，ここでは前者の方法を考えてみた。しかしばあいによっては後者のような方法を考えなければならないであろう。

最後に，カウント方式においてもひとつの試みを行なってみたことを言い添えたい。イギリスのフェランティ・マーキュリを用いた語彙調査（「国文学」昭和38年1月臨時増刊 147ページの水谷静夫氏の記述参照）ではデータのひとつひとつをテーブル・ルック・アップしてそこへ1ずつカウントするという方式をとっているようである。あるいはメモリーの大きさの関係かもしれない。もし補助記憶装置があればこの実験のように全データをソートして同じものをひとところに集め，これをカウントするという方式をとった方が早くできるのではないだろうか。この実験ではこのようないわばソート・アンド・カウントの方式を用いてみた。

5　終わりに

このプログラムはメインルーチンと七つのサブルーチンからなり，メイン・ルーチンは 355 ステップ，サブルーチン平均約百ステップで合計約千ステップからなる。プログラムの設計とコーディングは1964年1月～3月を要した。機械によるデバッグは同年8月31日～10月3日の間に行なった（延べ18時間）。10月6日公開実験を行なった。

実験に当たっては日本ビジネスコンサルタントの多くの方々にお世話になったが，特に中田宏，伊藤祐太郎両氏には長い時間にわたって機械のオペレートをしていただいたり，デバッグの相談に乗っていただいたりした。ここにお礼を申しあげる。