B 2. 辞 書 フ ア イ ル

坂本義行，蓼沼良一 （電気試験所）

目 次

ま え が き

機械翻訳用の辞書は，一般に，原語側と訳語側とに2分され，それぞれの側が，形態的情報，構文的情報，意味的情報の3つのブロックから構成されており，それぞれのブロックによって異なった構造を示している．したがって，これを記録するのに，すべての部分を同一構造で記述するのは，大容量ファイルの処理という点から，必ずしも得策ではない．

ここに報告するのは，辞書作成，使用の内面から，

1. 作成の難易
2. 記 憶 容 量
3. 検索時間（回数）
4. 内容の追加，削除，変更の可否

などの各項を考慮した能率的な辞書構造について述べたものである．

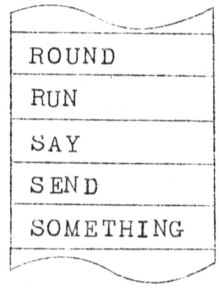
1 レコードの型

ファイルは，COBOL流に考えると，Uコードの集合ということができる．したがって，集合「ファイル」について述べるときには，その要素である「レコード」と，レコードがどのように組合さっているかという「ファイル」の構造とについて説明することが必要であろう．

以下には，要素であるレコードの構成と，これによって辞書を作るためのファイルの構造とのいくつかについて述べよう．

1.1 直線型レコード

レコードは，固定長あるいは可変長のもので，レコードには記憶すべき情報のみが記録される．レコード型としては，最も簡単なもので，このレコードによるファイルの構成は，第1図に示すように，直線状になる．

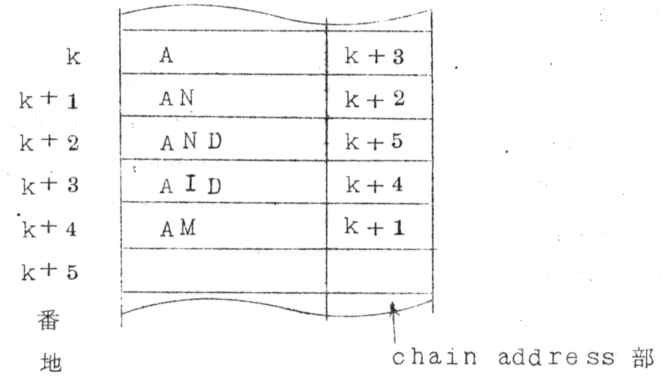


第1図　直線型ファイル

このファイルでは，一般には，読み込み，検索は順次に行い，情報のみを扱えばよいから，簡単である．しかし，情報の挿入，削除，修正などは，作り直しになるので，面倒になる．

1.2 チエーン・リスト型

この型は，第2図に示すように，直線型のレコードに，chain addressを加えた形のレコードである．



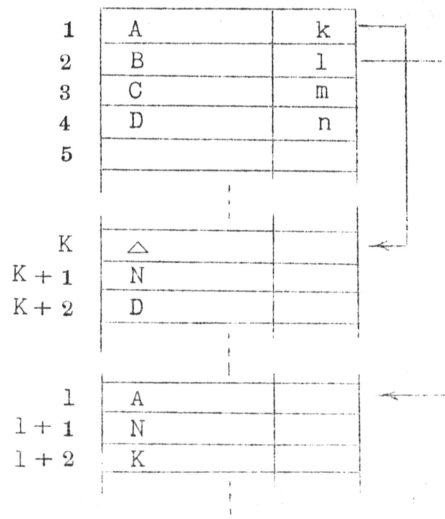
第2図　チエーン・リスト型ファイル

したがって，直線型に比べ，挿入，変更が可能となる．それは，chain addressの変更のみによって行える．しかし，ファイルの容量は，chain address分だけ増加するという不利な点が生ずる．

1.3　テーブル型

チエーン・リスト型のレコードでは，レコードの長さ以上のデータを記録できない．そこで，データは，それを構成する要素に分解し，レコードも要素だけを記録するようにする．こうすることにより，可変長のデータも効率よく扱える．

このレコードによるファイルの構成は，第3図に示すようになる．たとえば，BANKという語は，2番地のBから，1番以下のレコードにつながる．

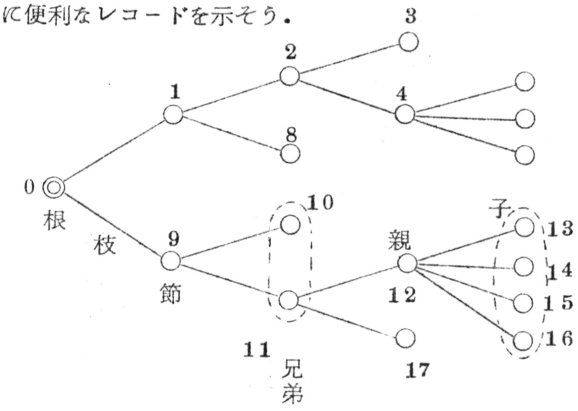


このファイルの構造は，階層構造しており，レコードの追加，削除がひんぱんに行われる場合には，かなり，有効な方法である．検索時間もまえの2つよりかなり早くなる．

第3図　テーブル型

1.4 樹 木 型

いろいろな情報のなかには，第4図のような木の型の構造をもつものが多い．そこで，この型の情報を扱うのに便利なレコードを示そう．



第4図　本 構 造 (1)

第4図に示すように，一つの「節」から出る「枝」の数は，一定していないのが普通である．「枝」の数が常に2本である「2本枝の木」や，常に3本である「3本枝の木」などでは，レコードは固定した形式のものですむ．

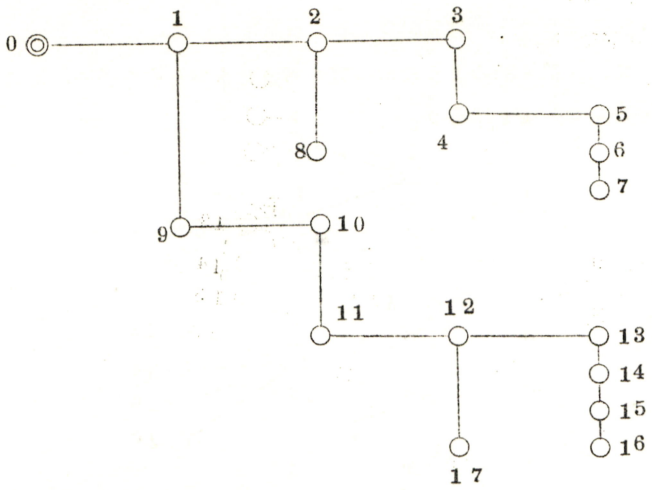
A	B	C	D	…

A：「節」のデータ
B：「節」の性質
C，D，…：「枝」

一般の場合では，レコードの形式が可変長になるという欠点がある．そこで，第4図の「木」を第5図のように変形する．このようにすると兄弟関係，親子関係を示す部分を用意したレコードで表わすことができる．したがって，いつも固定したレコードによって，どんな木でも表わせるようになる．

A	B	C	D

A：「節」のデータ
B：「節」の性質
C：兄弟関係
D：親子関係



第 5 図　木　構　造　(2)

この型のレコードは，4 つの部分から構成されるのが基準であるが，データの性質により，いろいろの変形が行われる．

2　辞書の性質

2.1　見　出　し

辞書は，「まえがき」で述べたように，語形，構文，意味の3種の情報を与えるものである．そこで，辞書作成にあたっては，語の「見出し」を決めなければならない．以下，翻訳用の英語辞書について考える．

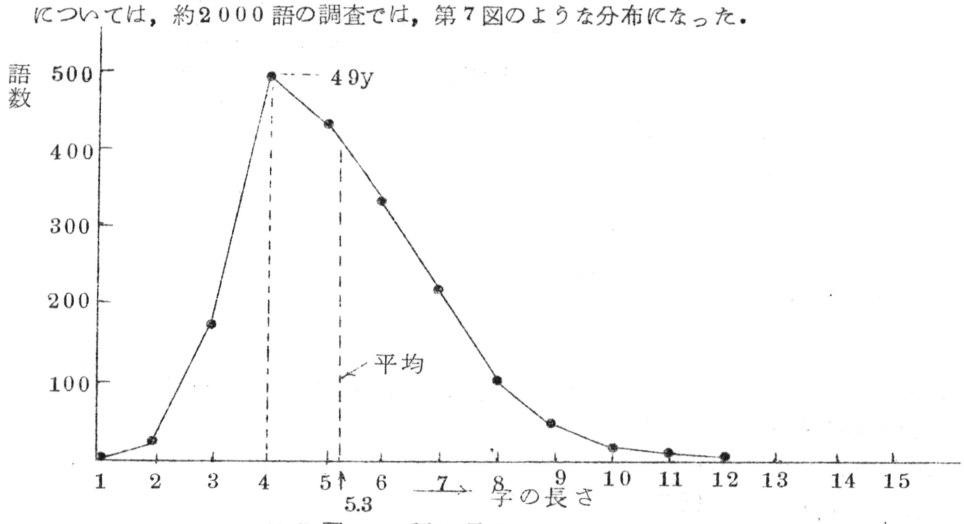
実際に，ある範囲で使われる英単語の数は，数十万語程度であろう．また，語の長さも 20 字前後までであろう．そこで，これらの単語を書き列べると，第6図のようになる．各列の文

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	A	B	L	E											
2	E	X	P	E	C	T	A	T	I	O	N				
3	L	O	A	D											
4	C	O	M	P	O	T	E	R							
5	C	O	M	M	O	N									
6	⋮														
⋮															

第 6 図　　見　出　し　表

字は，26種（大小を区別しないで）で，タテ方向に，極度に長いものになる．なお，語長に

については，約2000語の調査では，第7図のような分布になった．



第7図　　語の長さ

第6図を左端からながめていくと，同じ字列を含むものがかなり多く認められる．また，右端からながめていっても，同じような現象が見られる．これらに着目すれば，語の整理がかなりできることは，周知の通りである．

一般に，語は形態論的にみると，

語＝接頭辞＋語幹＋接尾辞

と分解することができる．すなわち，語は接頭辞，接尾辞を分離し，残りを「見出し」として辞書に入れればよい．この場合も，本当に語幹を見出しにするものと，原形を見出しにするものとがある．たとえば，lovingの原形はloveであるが，語幹としてlov をとるというようなものである．

また，少数の語に対しては，形態上から，規則的分解が容易でない，不規則変化語を別途に処理する必要が生ずる．これについては省略する．

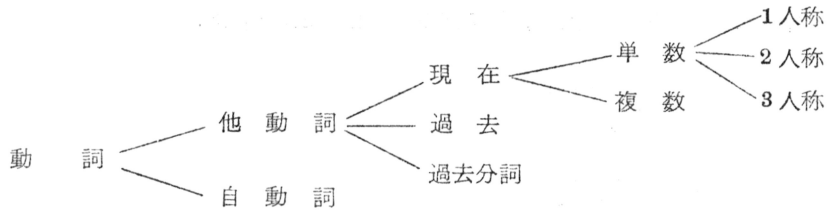
2.2　品　詞

品詞は，辞書がもっている構文情報の1つである．通常，語に対して与えられているものであって，少なくとも1ケ持っている．これに対して，複数個の品詞を持っている語を多品詞語と呼ぶ．

例えば，「up」のように，1語で，副詞，前置詞，形容詞，名詞，動詞と5個の品詞を所有するものである．約2000語の見出しに対する調査によれば，1語あたり1.7個という結果が得られた．

品詞もまた，階層構造によって，第8図のように，細分類して与えることもできる．

この他に，句，節に対しての構文情報を与えることも考えられる．



第　8　図　　品詞の階層構造

2.3　屈折語尾

屈折言語（Inflextional Language）と呼ばれる英語などにおいては，その構文情報を，その形態上に語形変化という形で与えている．

屈折語尾とは，ed,ing 等の例でわかるように，品詞の細分類された情報を与えるもので，人称，格，数，比較等を，1連の文字列の右端に一定の字列を加えたものである．

その種類ならびに，見出しとの結合頻度を第1表に示した．この屈折語尾の分離によって，見出し数を十分に減少させることが可能となる．

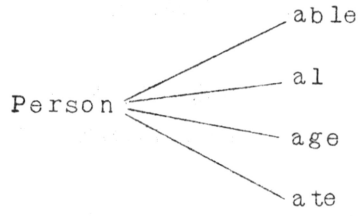
語　尾	S	ES	ED	'S	'	ER	EST	EN	ING	計
見出し数	1368	169	962	1326	33	511	142	71	1041	5623

第　1　表　　屈折語尾接続頻度表

しかし，分離に際して，その分離点での処理という問題が加わるという不利な面が加わる．

2.4　派生語尾

派生語尾と呼ばれているのは，語に与えられている構文情報である原品詞を，一定の字列を加えることによって，形態上で，他の品詞へ，すなわち，構文情報の変換を行う役割をもっているものである．この点で，屈折語尾と，情報面での役割を異にしている．その例を第9図に示し，代表的な種類と見出しとの接続頻度を第2表に示す．



第　9　図　　派生語尾接続例

派生語尾	個　数	派生語尾	個　数
ly	271	ant	21
ness	149	ent	10
less	155	age	37
y	175	th	25
able	135	ous	29
ful	83	ship	29
al	69	ic	17
ion	46	ical	10
ism	46	hood	20
ize	55	ate	16
ive	59	ary	11
ish	60	ure	10
ment	46	dom	10
ity	37	ess	10
ty	13		
ance	27		
ence	10		
or	38	計	1729

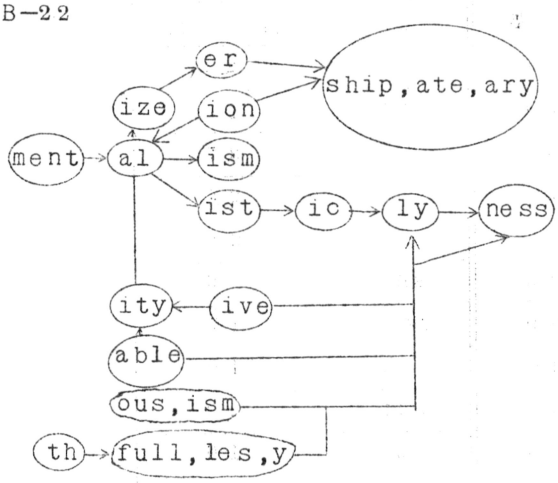
第　2　表　　派生語尾接続頻度表

ここで，特に注意したいのは，屈折語尾は1見出しに，同時に，1種の語尾，1回限りの接続のみが許されるが，派生語尾は，第10図の例でわかるように，数種をつぎつぎに接続できるという点で，屈折語尾と異った接続関係を有している．

```
o accident + al + ly
o posit + ive + ness
o real + ist +ic + al + ly
```

第10図　　複合派生接続例

．このような複合派生語尾間の接続関係には第11図に示したような関係がある．



第11図　派生語尾接続関係

3　最適レコード型の評価

3.1　レコード型の比較

第1章で述べた各レコード型の比較を，次の各項目に対して，概略比較を行ったものを第3表に示した．

1. 記憶容量（要素×素子数）
2. ローデイングの難易
3. 検索法
4. 検索回数
5. レコードの長さ
6. 変更手続

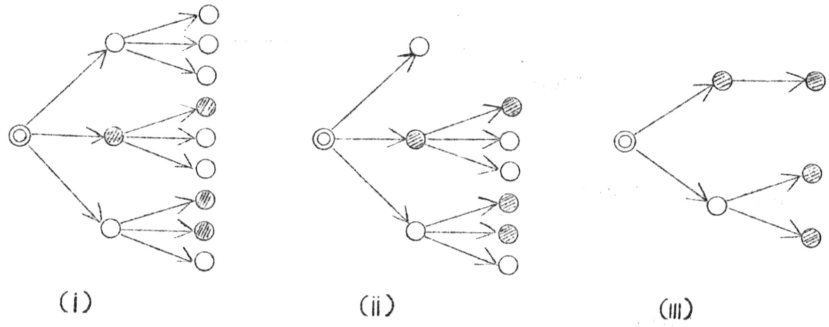
レコード型	1	2	3	4	5	6
直線型	大	○	順次	大	固定	不
チエーン・リスト	大	○	アドレス	大	固定	可
テーブル	中	×	アドレス	中	半固定	可
トウリー	小	×	アドレス	小	可変	可

第3表　レコード型比較表

第3表で，辞書ファイルに適するレコードを選択するには，1,4,5の各項が重要な点である．なお，チエーン・リスト，テーブル型は，トゥリー型の変形とも考えられるので，トゥリー型について検討を行うことにする．

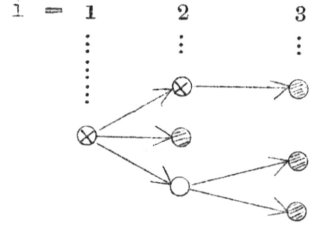
3.2　トゥリー構造

トゥリー構造を表現するには，第12図に示す3通りがある．ii)はi)で，子のない節をうけない場合であり，iii)は，　再に終端の節は，すべて葉の節から成り立っている場合である．



第12図　　トゥリー表現

iii) 図を次の13図の如く表現し，その各機能および個数を第4表に示す．



第13図　　トゥリー構造

シンボル	iレベルの節の数	機能	
		葉	接続
⊗	A_i	○	○
◎	B_i	○	×
○	C_i	×	○

第　4　表　　シンボルの機能

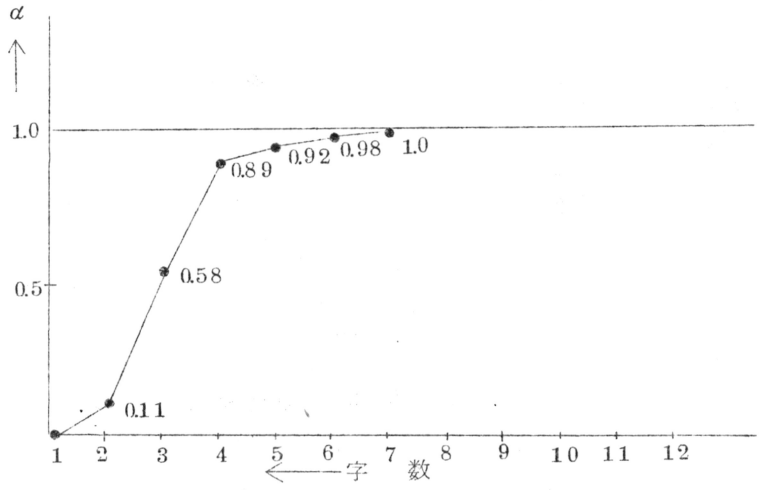
iレベルの節の全数をF_iで表わすと，

$$F_i = A_i + B_i + C_i \qquad (1)$$

で表わすことができる．またレベルiが葉の節であって，終端となる割合をα_iで表わすと，

$$\alpha_i = \frac{\text{iレベルで終端を示す葉の節}}{\text{iレベルのすべての節の数}} = \frac{B_i}{A_i + B_i + C_i} \qquad (2)$$

2000語の調査結果による α_i のグラフは，第14図に示すようなものとなった．この結果から，語長が8字程度で，$\alpha_i \fallingdotseq 1$，すなわち，$A_i \fallingdotseq 0$ となることがわかる．これは，葉兼接続の節が極端に少なくなり，チエーン型に連なった語形を示すことがわかる．



第14図　葉の節で終る割合 α_i

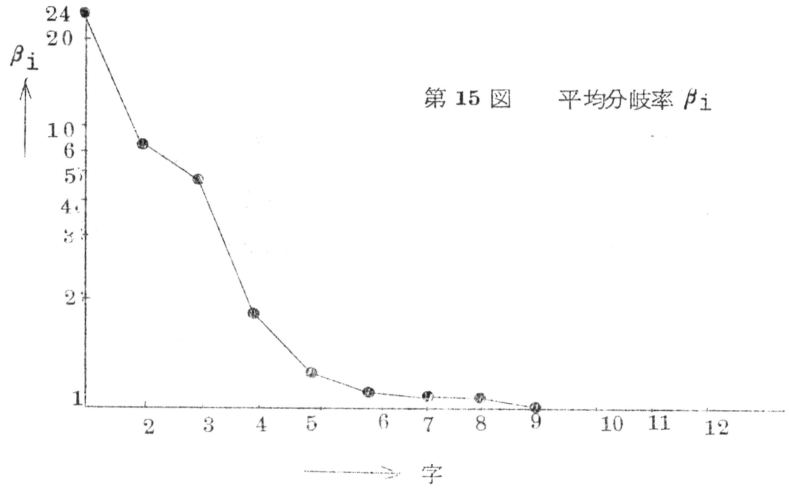
次に，レベルiでの平均の分岐率を β_i とすれば，

$$\beta_i = \frac{\text{レベルiの節}}{\text{レベルiに節を出す節}} = \frac{F_i}{F_{i-1} - \beta_{i-1}} \qquad (3)$$

(3)式より，語長5字程度で $\beta_i = 1$ となり，(2)式の結果以上に，チエーン・リスト型のデータ構造を示すことがわかる．その調査結果を第15図に示す．この β_i を用いて，i＋1レベルの節の数を，

$$F_{i+1} = \beta_i (F_i - \alpha_i \beta_i) \qquad (4)$$

の式を用いて，ある程度推定することが可能である．



第15図　平均分岐率 β_i

3.3　計算機でのリスト構造

i)　4型（標準型）

この型は，第16図に示すように，A，B，C，Dの要素をすべて持っており，節の総数f_4はトウリー構造での節数に等しい．

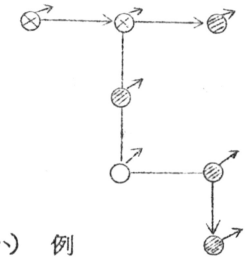
$$f_4=\sum_{i=1}^{n}F_i=\sum_{i=1}^{n}(A_i+B_i+C_i) \qquad (5)$$

A	B	C	D

(イ) 要　素

A	○	値
B	↓	弟
C	→	子
D	↗	定　義

(ロ) 機能表示



(ハ)　例

第16図　　4　型　の　節

ii)　3.5型

この型は，要素Cが2個のアドレスを表わし，指標部Dで識別を行う．その結果，子と定義を同時に共有できないため，定義を表わす空の節を必要とする．節の総数は$f_{3.5}$は，

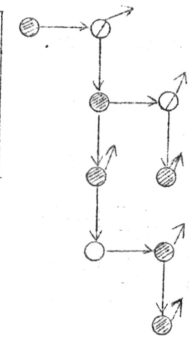
$$f_{3.5}=\sum_{i=1}^{n}(F_i+A_i) \qquad (6)$$

A	B	C	D

(イ) 節の要素

A	○	値
B	↓	弟
C	→|↗	子1定
D	0|1	0 子，1 定

(ロ) 機能表示



○：共有のための増加節

(ハ)　例

第17図　　3.5 型の節

iii)　3　型

3.5型と同じく，要素Cが交代するのであるが，指標部をもうけず，Aの要素を識別用に兼

用する．そのため，Aが空でない節の定義を行うための節の増加を必要とする．その図を第18図に示し，節の総数f_3は，

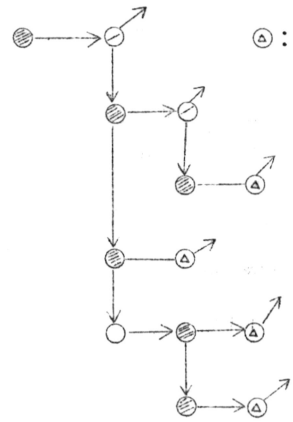
$$f_3 = \sum_{i=1}^{n} (F_i + A_i + C_i) \qquad (7)$$

A	B	C

A	○	値 | 空
B	↓	弟
C	→ | ↗	子 | 定

(イ) 節の要素

(ロ) 機能表示



Ⓐ：Aが空でないための増加節

(ハ) 例

第18図　3型の節

iv) 2.5型

これは，テーブル型とも考えられ，弟，子のいずれかが，直線型で示される場合であって，その容量は，ほぼ3.5型に等しくなるが，効率のよいローデングが非常に困難である．その節の総数$f_{2.5}$は，

$$f_{2.5} = \sum_{i=1}^{n} (F_i + A_i) \qquad (8)$$

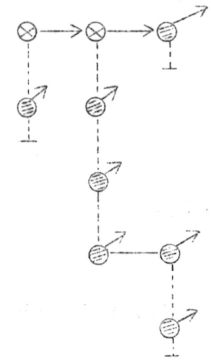
で示され，その節の図を第19図に示した．

A	B	C

(イ) 節の要素

A	0	値 | 空
B	↗ →	子 | 定
C	0 子－ 定－ 1	接続
	2 子－ 定－ 3	終端

(ロ) 機能表示



：テーブルの終端を示す．

(ハ) 弟が直線型の例

第19図　2.5型の節

V) 2　型

2．5型におけるB，Cの識別をAで行うため，Bを共有する分と，各テーブルの終端で，各1個づつの節の増加を必要とする．即ち節の総数f_2は(9)式で示され，その図を第20図に示した．

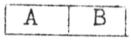
$$f_2 = \sum_{i=1}^{n} (F_i + F_{\sigma i} + F_{\delta i}) \qquad (9)$$

ここで，

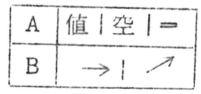
$$F_{\sigma i} = F_i - C_i$$

$$F_{\delta i} = F_{i-1} - C_{i-1}$$

とする．

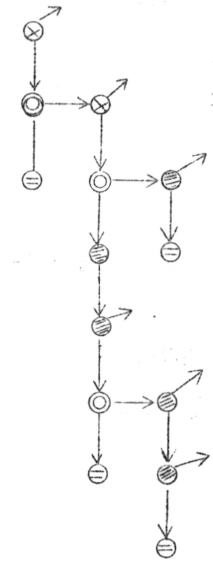


(イ) 節の要素



(ロ) 機能表示

◎＝$F_{\sigma i}$
⊖＝$F_{\delta i}$
の節を表わす．



(ハ) 例

第20図　2型の節

3．4　記憶容量の比較

前節で分類したI)～V)の型について記憶容量の比較を行ってみる．

ここで，各型の節の容量をCで表わせば全容量Mはそれぞれ，

$$M_4 = C_4 \times \sum_{i=1}^{n} F_i = C_4 \times \sum_{i=1}^{n} (A_i + B_i + C_i) \qquad (10)$$

$$M_{3\,5} = C_{3\,5} \times \sum_{i=1}^{n} (F_i + A_i) = C_{3\,5} \times \sum_{i=1}^{n} (2A_i + B_i + C_i) \qquad (11)$$

$$M_3 = C_3 \times \sum_{i=1}^{n} F_i = C_3 \times \Sigma (2A_i + B_i + 2C_i) \qquad (12)$$

$$M_{2.5} = C_{2.5} \times \sum_{i=1}^{n} (F_i + A_i) = C_{2.5} \times \sum_{i=1}^{n} (2A_i + B_i + C_i) \qquad (13)$$

$$M_2 = C_2 \times \sum_{i=1}^{n} (F_i + F_{\sigma i} + F_{\delta i}) = C_2 \times \sum_{i=1}^{n} (3A_i + 3B_i + C_i) \qquad (14)$$

と表わすことが出来る．

次に，見出し2000語の辞書に要する記憶容量の算定を行った．

節の容量

4 型	:	X(1) | 9(4) | 9(4) | 9(4)	$C_4 = 13$ ケタ
3.5型	:	X(1) | 9(4) | 9(4) | 9(1)	$C_{3.5} = 10$
3 型	:	X(1) | 9(4) | 9(4)	$C_3 = 9$
2.5型	:	X(1) | 9(4) | 9(1)	$C_{2.5} = 6$
2 型	:	X(1) | 9(4)	$C_2 = 5$

ただし，

X＝英文字

9＝数字

( )内：ケタ数　　1ケタ＝6ビット

を表わすものとする．

(a) 見出しの節の数は：

$$\sum_{i=1}^{n} F_i = 5{,}000 \qquad \sum_{i=1}^{n} C_i = 1{,}700$$

$$\sum_{i=1}^{n} A_i = 200 \qquad \sum_{i=1}^{n} F_{\sigma i} = \sum_{i=1}^{n} F_{\delta i} = 3{,}300$$

(b) 品詞（1項品詞のみ）の節の数は

$$\sum_{i=1}^{n} F_i = 3{,}500 \qquad \sum_{i=1}^{n} C_i = 1{,}500 \qquad \sum_{i=1}^{n} B_i = 0$$

$$\sum_{i=1}^{n} A_i = 2{,}000 \qquad \sum_{i=1}^{n} F_{\sigma i} = \sum_{i=1}^{n} F_{\delta i} = 2{,}000$$

(c) 派生（2次派生のみ）の節の数は

$$\sum_{i=1}^{n} F_i = 7{,}500 \qquad \sum_{i=1}^{n} B_i = 0 \qquad \sum_{i=1}^{n} F_{\sigma i} = \sum_{i=1}^{n} F_{\delta i} = 2{,}000$$

$$\sum_{i=1}^{n} A_i = 2{,}000 \qquad \sum_{i=1}^{n} C_i = 5{,}500$$

(d) 屈折の節の数

$$\sum_{i=1}^{n} F_i = 7500 \qquad \sum_{i=1}^{n} A_i = \sum_{i=1}^{n} B_i = \sum_{i=1}^{n} F_{\sigma i} = \sum_{i=1}^{n} F_{\delta i} = 0$$

$$\sum_{i=1}^{n} C_i = 7500$$

以上のデーターを基にして計算を行った結果第5表に示した.

単位 1000ケタ

型 空間	見出し	品詞	派生	屈折
M_4	65	45.5	97.5	97.5
$M_{3.5}$	52	55	95	75
M_3	75	63	135	135
$M_{2.5}$	31	33	57	45
M_2	58	38.5	57.5	37.5

第 5 表 記憶容量比較表

3.5 検索時間

時間の比較は計算機, プログラム等によって異なるものと思われる. そこで基準となる次の3つの項目によって, 概略するにとどめた.

i) 各節の値を比較し, 番地へ飛ぶに要する平均時間を ta,

ii) 番地の識別の指標に要する時間を tb

iii) 各節の値を比較し, 隣の番地へ飛ぶに要する時間を tc

と仮定すると, 見出しに対して

4 型: 平均語長を5字とすると, 各レベルの節の数に比例する. 各レベルの節の数は最大267であるから,

$$T_4 \leqq \frac{1}{2} ta \times 26 \times 5 = 55\,ta$$

3.5型: 各レベルの節の数が最大27ケとなるから,

$$T_{3.5} \leqq \frac{1}{2}(ta + tb) \times 27 \times 5 = 67.5(ta + tb)$$

3 型: この場合は, 検索出来た後, 一節だけ多く参照する結果

$$T_3 \leqq (67.5 + 1)\,ta = 68.5\,ta$$

2.5型: 節の数は3.5型に等しいから

$$T_{2.5} \leqq \frac{1}{2}(tc + tb) \times 27 \times 5 = 67.5(tc + tb)$$

2 型: 最大の場合節の数が各レベルで, 接続の役割を果す1個の増加が考えられるので,

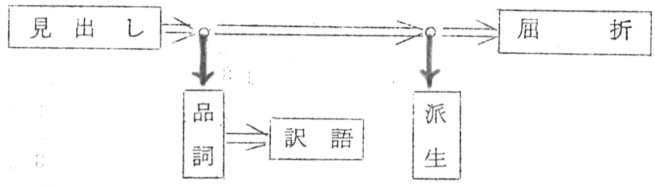
$$T_2 \leqq \frac{1}{2} tc \times 27 \times 5 = 67.5\,tc$$

となる．

なお，品詞，派生，屈折については省略する．

4　辞書ファイルの構造

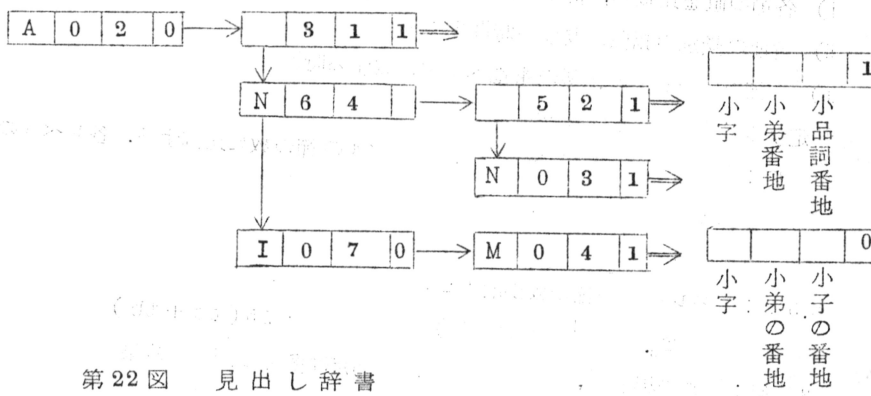
辞書の全体の空間構成を第21図の如くに決め，各空間のレコード型は3章での検討から，最適レコードを決定した．



第21図　辞書の構成

4.1　見出しレコード

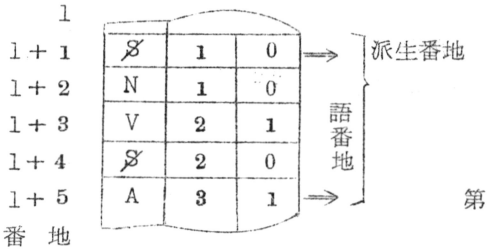
3.5型を使用することとした．この型では，ローデングにおいて，A_i の処理を必要とするため，これが容易に行えるよう，アルファベット順にロードすることとした．第22図は，その構造例を示したものである．



第22図　見出し辞書

4.2　品詞レコード

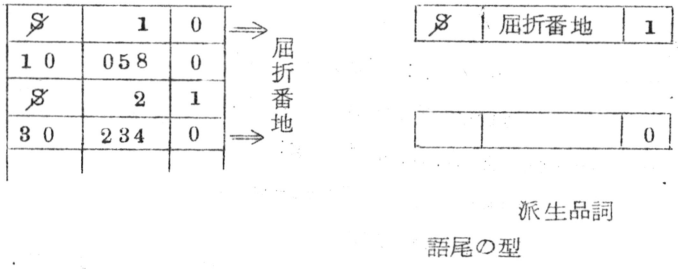
1次品詞のみを採用することとしたので，検索時間を短縮し，かつ高度の可能性を考慮して，2,5型を用いることとした．その構造例を第23図に示した．この場合，派生空間への接続のため，各見出しに1個づつの割合で節をもうけた．



第23図　品詞辞書

4.3　派生レコード

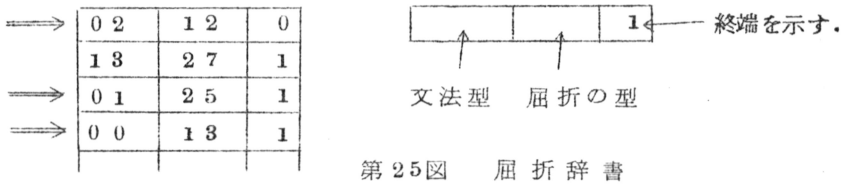
レコード型は，品詞と同様に，1次派生のみを採用することをして，2.5型を用いた．その構造および要素の機能を，第24図に示した．



第24図　派生辞書

4.4　屈折レコード

レコード型は，派生と同じものを用いた．ただし，屈折に対しては，接続節を必要としない点が，派生と異なっている．その構造および機能を第25図に示した．



第25図　屈折辞書

あ と が き

辞書ファイルの形式を，いくつかの空間に分解し，最適と思われるレコード型を求めたわけだが，あまりに細分化することは，空間の間での互換性を少なくさせ，辞書引きにおけるプログラムは，非常に複雑なものとなる．ここでは省略したが，将来は，品詞，派生語尾をも階層構造をとり入れる必要があろう．なお訳語（日本語）側については，日本語形の特殊性から，現在検討中である．

辞書容量の増加にともない，外部記憶装置，特に磁気テープへの記録に対しては，別の問題

があると思われる。

実際にこの辞書作成に用いたのは，

計算機：FACOM230-50，NEAC-2200

入　力：カード

（プログラム，データー）

出　力：磁気テープ

プログラム：コボル・コンパイラー

を用いた。

最後に，英語の調査資料を提供下された，玉井氏，又プログラム作製で御援助下された，研究室の諸氏に深謝する次第です。

参　考　文　献

(1) E.H.Sussenguth Jr.: Use of Tree Structure for Proccesing Files Com. of ACM 1963 vol 6

(2) A.K.Scidmore & B.L.Weibag: Storage and Search Properties of a Tree-Organijed Memory System, Com. of ACM

(3) 玉井陽子：英語辞書－見出しの木構造　MT委員会 July. 1966

(4) 玉井陽子：英語辞書－見出し　MT委員会 May. 1966

(5) 西村怒彦：木表現とリスト処理算法　MT委員会 Jan. 1966

(6) 坂本義行・蓼沼良一：英語定形辞書　MT委員会 Sept. 1967

本 PDF ファイルは 1968 年発行の「第 9 回プログラミング一シンポジウム報告集」をスキャンし、項目ごとに整理して、情報処理学会電子図書館「情報学広場」に掲載するものです。

この出版物は情報処理学会への著作権譲渡がなされていませんが、情報処理学会公式 Web サイトの https://www.ipsj.or.jp/topics/Past_reports.html に下記「過去のプログラミング・シンポジウム報告集の利用許諾について」を掲載して、権利者の捜索をおこないました。そのうえで同意をいただいたもの、お申し出のなかったものを掲載しています。

過去のプログラミング・シンポジウム報告集の利用許諾について

情報処理学会発行の出版物著作権は平成 12 年から情報処理学会著作権規程に従い、学会に帰属することになっています。

プログラミング・シンポジウムの報告集は、情報処理学会と設立の事情が異なるため、この改訂がシンポジウム内部で徹底しておらず、情報処理学会の他の出版物が情報学広場 (=情報処理学会電子図書館) で公開されているにも拘らず、古い報告集には公開されていないものが少からずありました。

プログラミング・シンポジウムは昭和 59 年に情報処理学会の一部門になりましたが、それ以前の報告集も含め、この度学会の他の出版物と同様の扱いにしたいと考えます。過去のすべての報告集の論文について、著作権者（論文を執筆された故人の相続人）を探し出して利用許諾に関する同意を頂くことは困難ですので、一定期間の権利者捜索の努力をしたうえで、著作権者が見つからない場合も論文を情報学広場に掲載させていただきたいと思います。その後、著作権者が発見され、情報学広場への掲載の継続に同意が得られなかった場合には、当該論文については、掲載を停止致します。

この措置にご意見のある方は、プログラミング・シンポジウムの辻尚史運営委員長 (tsuji@math.s.chiba-u.ac.jp) までお申し出ください。

加えて、著作権者について情報をお持ちの方は事務局まで情報をお寄せくださいますようお願い申し上げます。

期間：2020 年 12 月 18 日 〜 2021 年 3 月 19 日
掲載日：2020 年 12 月 18 日

プログラミング・シンポジウム委員会

情報処理学会著作権規程
https://www.ipsj.or.jp/copyright/ronbun/copyright.html