解　説

自然言語理解の基礎—形態論†

日　高　　達††

1. はじめに

形態論は語における語形変化 (inflection), 派生語 (derivation), 合成 (compounding) の構造を取り扱う. たとえば, 基本形の動詞 reach は, reaches, reached, reaching と語形変化し, 種々の接辞 (Affixes) をともなって reachable, reacher, unreachable などの派生語をつくり, 他の語と結合して reach-me-down などの合成語をつくる. また, 名詞 book は books と複数形の語形変化をし, 接辞をともなって派生語 bookish, booklet をつくり, 他の語と結合して bookcase, bookend, bookleaned, bookman などの合成語をつくる.

語形変化は基本形の語の性や可算性, 品詞, 末尾の音形により, 多くの場合, 規則的な変化をするが, 接辞の接続は基本形の語の意味範疇に依存して接続したりしなかったりするし, 合成にいたっては合成語を構成する個々の語に強く依存して合成語として自立したりしなかったりする. たとえば,「細い」と「面」の合成による「細面」は語になっても,「細い」の対語である「太い」と「面」の合成による「太面」は語にならない. このようなことから, 合成語の規則は整理されるにいたっておらず, 語形変化と派生の規則が形態論で整理されている. したがって, 形態論は辞書に登録された見出し語 (基本形の語と合成語) の集合と, その語形変化規則および派生語生成規則によって表される.

自然言語の機械処理では語形変化や派生語として出現した語を認識する形態素処理が必要であるが, 形態素処理は言語により, また用いられる語彙によりかなり異なる. たとえば, 英語ではきわめて単純な語形変化をし, 派生語の形成力もそれほど強くない. したがって, 語彙を制限した自然言語処理では変化形や派生語をすべて機械辞書に登録しておくことにより, 形態素処理を行わない. しかし, 英語においても, 派生語の発音は音韻学的に複雑で, 音声認識や音声合成では慎重な取り扱いが必要である. 一方, 独語やロシア語では複雑な語形変化をし, 特にフィンランド語では名詞が2千余の語形変化をし, 動詞は1万2千もの変化形をもつという. このような言語では, すべての変化形や派生語を機械辞書の見出し語として登録しておくことは実際的でなく, 形態素処理を省略するわけにはいかない.

形態論における語の概念は必ずしも単語の範囲にとどまるものではなく, 日本語の場合には文節を語として取り扱うことができる. 文節は一つの自立語に複数個の付属語 (助動詞, 補助用言相当語句, 助詞) が接続したものであるが, その接続には一定の規則性があり, 付属語を接辞と考えて, 文節を形態論の立場から取り扱うことができる.

日本語は, 単語単位で分かち書きされる英語などと異なり, 一般にべた書きされる. したがって, 日本語の機械処理では文字列として入力された文から単語を認識し, 単語列に変換する操作が必要となる. 日本語では同音異義語が多いことから, 単語列に変換する際に多数の曖昧さが発生するので, 文節における単語連接の拘束性 (文節文法) を用いて曖昧さを減少させる. この技術は, 昭和42年頃から「かな漢字変換」の基盤技術として発達し, 今日のワードプロセッサの普及に貢献した. 本稿では, まず単語における形態論を概括し, 日本語における形態素処理と機械辞書のデータ構造について説明する.

2. 形　態　論

形態素とは, 語を構成する有意義最小言語形式であり, 形態論では, 語の形成を形態素の結合操作として捉える. この場合, 結合操作とは, 形態素の線形結合である語における形態素とその直後の形態素の間の連接可能性である. 一般に, 形態素は語基 (または語

† Basic of Natural Language Understanding—Morphology by Toru HITAKA (Department of Electronics, Faculty of Engineering, Kyushu University)

†† 九州大学工学部電子工学科

幹）と接辞，助辞（屈折辞）から成る．

語基は語の基幹を成す形態素であり，品詞またはその下位範疇に大分類される．語基は“本”のように単独で自立して語になる自立型と，“走（はし）”のように他の形態素と結合して語を形成する結合型がある．

接辞は語基について自立型の（派生）語基を形成する．たとえば，接尾辞 -ment は動詞について動作またはその結果・手段を表す名詞をつくり (N→V•ment)，接頭辞 un- は形容詞や副詞について否定の意味を付加する (A→un•A，Ad→un•Ad)．日本語の文節を語として捉えると，用言に付く助動詞は接尾辞とみなすことができる．これらの助動詞は用言に接続して態 (voice) や時制 (tense) などを表す．

助辞は語基について屈折語を形成する．ここで，屈折 (inflection) とは文中におけるさまざまな文法関係を表すために語が形を変えることをいい，名詞・代名詞の格・性・数にともなう語形変化や，形容詞・副詞の比較変化 (comparison)，動詞の活用語尾などである．日本語の文節では助詞がこれに相当する．以上のことから，語は一般に次のような形態をとる．

語＝(接頭辞)・語幹・接尾辞*・(助辞)　(1)

語の構成は(1)の形式であればなんでも語になるわけではなく，一定の規則性がある．この規則性は(1)において，形態素とその直後の形態素の接続上の拘束である．この拘束は，3.で述べるように，正規文法として記述される．

形態素の接続拘束性に基づき語の構成を解析する処理を形態素解析という．形態素の接続拘束性が正規文法であり，正規文法の解析は $O(n)$ の能率のアルゴリズムが広く知られているから，形態素解析は $O(n)$ の能率で実行可能である．ただし，n は語の文字長さである．

語と語の間にスペースを置き，語単位で分かち書きされる欧米語では，形態素の接続拘束性が単純で，語の文字長さも短く，形態素解析はほとんど問題ないという者が多い．しかし，日本語や中国語のようにべた書きされる言語では，長い文字列としての入力文から，単語を認識して，単語列へ精度良く変換する処理が必要であり，依然として機械処理上の基本問題である．

3. 日本語の形態素解析

日本語は，語単位で分かち書きされる欧米語などと異なり，複数文節単位で分かち書きされる．したがって，日本語の構文解析では，文字列としての文節列を単語の並び（単語列）に変換する特殊な工程を必要とする．文字列を単なる単語の列として解析すると，多くの曖昧さが生じる．特に，入力文が仮名またはローマ字表記の場合には，日本語として不適当な解析結果が生ずる．

例　入力文字列：にわにはにわとりがいる．

1　荷・ワニ・羽・に・輪・と・利・が・要る．

2　庭・に・は・鶏・が・いる．

したがって，文節列としての文字列の上で成立する拘束条件を用いて，単語列に変換する必要があるが，この拘束条件として，一般に「文節文法」が用いられる．日本語の機械処理では，文節文法により，文字列を単語列に変換する処理を形態素解析と呼んでいる．

3.1 文節文法

文節は，自立語に複数個の付属語（助動詞，補助用言相当語句（“ある”，“いる”，“かける”，“はじめる”など），助詞）が連接したものであり，次のような形態をとる．

文節＝自立語・(助動詞)*・(助詞)*

ここで，＊は0個以上の連鎖を表す．

文節における助動詞の連接順序は一般に次のようになる．

助動詞*＝(使役)・(態)・(相)・(否定)・(時制)・(ムード)

ここで，使役，態，相，否定，時制，時，ムードは，以下に述べる助動詞の意味分類であり，( )は類に属する助動詞が0個または1個接続することを意味する．

使役．使役の助動詞“せる”，“させる”．動詞（自立語）の活用語尾の最後の音が「i」または「a」の強変化語尾の場合には“せる”が用いられ，弱変化語尾の場合には“させる”が用いられる．

態．受身の助動詞“れる”，“られる”．強変化語尾には“れる”が，弱変化語尾には“られる”が付く．尊敬や自発，可能の助動詞“れる”，“られる”もこの類に入れる．

相．事象の成立している時間の中で，着目する時点がどの位置にあるかを表す補助用言相当語句“かける”，“はじめる”，“ている”，“つつある”，“つづける”，“終る”，“てしまう”など．

否定．否定の助動詞“ない”．

時制．過去の助動詞“た”．

ムード．話者の意見を表現する助動詞“う”，“よう”，

“まい”，“らしい” など．

文節における単語列 $w_0 \cdot w_1 \cdot \cdots \cdot w_l$ では，語 w_k $(k=0, \cdots, l)$ とその直後の語 w_{k+1} には連接上の拘束条件が成立し，w_k $(k=0, \cdots, l)$ の品詞と活用形をそれぞれ H_k, K_k と記すと，拘束条件は H_k, K_k, H_{k+1} の述語Cとして表すことができる．

$C(H_k, K_k, H_{k+1})$

この場合，自立語の品詞は名詞，数詞，形容詞，副詞，活用の種類別の動詞，連体詞，またはその下位範疇であり，活用形は未然，連用，終止，連体，仮定，命令の各活用形と考えてよい．また，付属語は一般に一語一品詞である．たとえば，助動詞“ない”は動詞や助動詞の未然形に接続し，形容詞“ない”は形容詞の“く”連用形や形容動詞の“で”連用形などに接続する．

C（動詞，未然，助動詞“ない”）
C（助動詞，未然，助動詞“ない”）
⋮
C（形容詞，“く”連用，形容詞“ない”）
C（形容動詞，“で”連用，形容詞“ない”）
⋮

また，文節の末尾の語は，未然形や仮定形ではありえないように 終結条件Eで制限を受ける．Eは，一般に品詞と活用形の二項述語 E(H, K) である．文節の先頭の語は自立語を要求するので，Hが自立語の品詞であることを J(H) で表す．入力文字列が有限個の文節列であることから，述語Cを拡張し，

$C(H_k, K_k, H_{k+1}) \vee (E(H_k, K_k) \wedge J(H_{k+1}))$

を改めて，$C(H_k, K_k, H_{k+1})$ と定義すると，単語列 $w_0 \cdot w_1 \cdot \cdots \cdot w_l$ が文節列であるためには，次の(i)～(iv)が成立することが必要であり，これが文節文法と呼ばれるものである．

文節文法
(i)　$J(H_0)$
(ii)　$C(H_k, K_k, H_{k+1})$ $(k=0, \cdots, l-1)$
(iii)　$E(H_l, K_l)$

文節文法は，正規文法 (regular grammar) である．このことは，文節文法が次のような右線形正規文法に変換できることから，明らかであろう．

正規文法としての文節文法
(i)　$S \rightarrow [H]$　…J(H)
(ii)　$[H] \rightarrow w_H \cdot [H, K]$
(iii)　$[H, K] \rightarrow w_K [H'] \cdots C(H, K, H')$
(iv)　$[H, K] \rightarrow w_K$　…E(H, K)

ここで，Sは文節列の開始記号として導入された特別の記号であり，[H], [H, K] は非終端記号，w_H は品詞Hの語の語幹の綴り，w_K は活用形Kの語尾の綴りである．活用形Kが（カ行五段活用などの）活用の種類の情報まで含んでいると解釈すると，Kから活用語尾の綴り w_K が一意に決まる．

文節文法の品詞や活用を学校文法における分類から離れて再整理し，文節文法を簡潔化する試みが佐野らにより行われている[7]．

3.2 解析アルゴリズム

n 長さの入力文字列が与えられたとき，入力文字列の形態素構造を出力する手続きを形態素解析アルゴリズムという．一般に，形態素の構造規則は正規文法である．正規文法の解析は時間，記憶容量がともに $O(n)$ の解析アルゴリズムが一般に知られている．

n 長さの入力文字列 $s = a_1 \cdot a_2 \cdot \cdots \cdot a_n$ の部分文字列 $a_{i+1} \cdot a_{i+2} \cdot \cdots \cdot a_j$ $(0 \leq i \leq j \leq n)$ を $s(i, j)$ と記す．解析に用いる機械辞書は語幹を見出し語，品詞と活用の種類を内容として収容しているものとする．また，活用の種類ごとの活用語尾の表と文節文法規則をファイルにもっているものとする．単語の正書法綴り w，その品詞 H，活用形 K の三組 (w, H, K) をその単語の単語構造と呼び，文節列としての単語列に対応する単語構造の列を文節構造と呼ぶことにする．

例　たんごと呼ばれる
文節構造＝(単語，名詞，—)(と，接続助詞“と”，—)(呼ば，バ行五段動詞，未然)(れる，受動・助動詞“れる”，終止)

$s(i, j)$ の単語構造が α であることを $W(i, j, \alpha)$ で表すと，機械辞書と活用語尾ファイルを検索することにより，入力文字列 s の位置 i から始まるすべての単語構造の集合 $WS(i)$ を一定時間以内に検索することができる．

$$WS(i) = \{(i, j, \alpha) \mid i \leq j \leq n, \\ WS(i, j, \alpha)\}$$

3項述語Pを次のように定義する．$s(0, j)$ が文節文法規則(i)，(ii)を満足する文節構造 γ をもち，かつ γ の最右の語が $W(i, j, \alpha)$ であるとき，そのときにかぎり，$P(i, j, \alpha)$ とする．したがって，$P(i, n, (W, H, K))$ であるような i, W, H, K が存在し，かつ E(H, K) のとき，そのときにかぎり，s は文節列を成すことになる．$P(i, j, (H, K))$ は次の手順で帰納的に構成できる．

述語 P の構成法

(i) $W(0, j, (w, H, K)) \wedge J(H)$
$\Rightarrow P(0, j, (w, H, K))$

(ii) $P(i, j, (w, H, K)) \wedge (j, k, (w', H', K'))$
$\in WS(j) \wedge C(H, K, H')$
$\Rightarrow P(j, k, (w', H', K'))$

述語 $P(i, j, \alpha)$ の構成は，まず(i)を実行し，$i=1, 2, \cdots, n$ の順に(ii)を繰り返し実行すればよい．単語の長さが一定値より小さいことを考慮すると，述語Pの構成は $O(n)$ の能率で実行できる．この構成法は，横型で上昇型の手法である．もちろん，縦型バックトラック方式の構成法も容易に組み立てることができるが，能率は落ちる．

述語Pの構成が終わると，Pからsの文節構造を生成する段階に移る．文節構造の生成は，Pの構成の逆過程である．

文節構造 BS の生成

(i) $P(i, n, (w, H, K)) \wedge E(H, K)$
$\Rightarrow BS := (w, H, K),\ I := i.$

(ii) $L(BS) = (w', H', K') \wedge P(i, j, (w, H, K))$
$\wedge I = j \wedge C(H, K, H')$
$\Rightarrow BS := (w, H, K) \cdot BS,\ I := i.$

ここでは，L(BS) は単語構造の列 BS における最左の単語構造を値とする関数である．BS の生成は，まず(i)を実行し，次に $I=0$ になるまで繰り返し(ii)を実行することにより，最終的にsの文節構造 BS をうる．文節構造生成の能率は $O(n)$ である．

3.3 ヒューリスティクスの利用

形態素解析では，特に入力文がかな文字列の場合，多くの曖昧さが発生する．その理由の一つは，文節文法が直前・直後の語の連接条件にすぎず，文節間の係り受けや意味処理を一切無視していることにある．しかし，形態素処理が構文解析の前処理であり，構文解析は本質的には文脈自由文法の構文解析で，$O(n^2)$ の能率であることを考えると，納得できることである．

形態素解析において発生するたくさんの曖昧さのなかから，正解らしい解を優先的に出力する方法に，最長一致法と文節数最小法がある．最長一致法は，欧米語の形態素解析において接辞の切り出しを行う際に，語末から最も長い接辞を優先的に切り出して接辞と語幹の分離を行う方法であった．我が国では，九大でかな漢字変換が始まった昭和42年の頃からすでに取り入れられており，集合 $WS(i)$ の元のなかで最も長い語を優先する方法である．文節数最小法は，文節数，つまり自立語の個数が少ない解のなかに正解がある場合が多いという経験に基づき，自立語の個数が最小な文節構造を優先的に出力する方式である．吉村らの報告によると，文節数最小法は最長一致法に比べて大幅な正解率の上昇が期待できることが報告されている．図-1～図-3 は最長一致法と文節数最小法との性能比較を示す．これは，武者小路実篤の「人生論」から最初の1,000文を平仮名表記のべた書き文に直したものを入力文（平均44文字）とし，自立語機械辞書は約83,000語，付属語機械辞書には約300語の付属

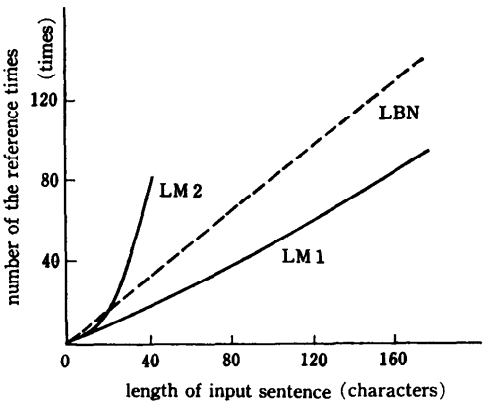


図-1 自立語辞書の検索回数
LM 1：最長一致法（第一番目の解析を求める場合），LM 2：最長一致法（正解を求める場合），LBN：文節数最小法．

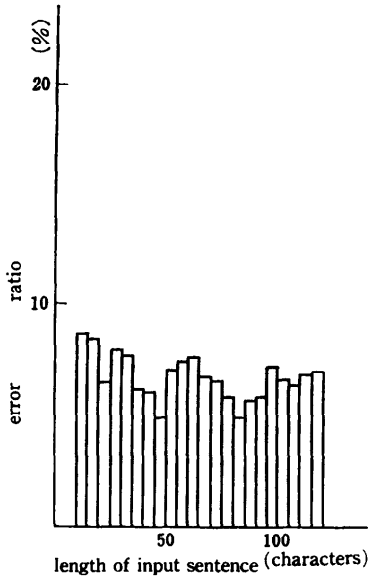


図-2 最初に出力された解析の誤り率（文節数最小法）

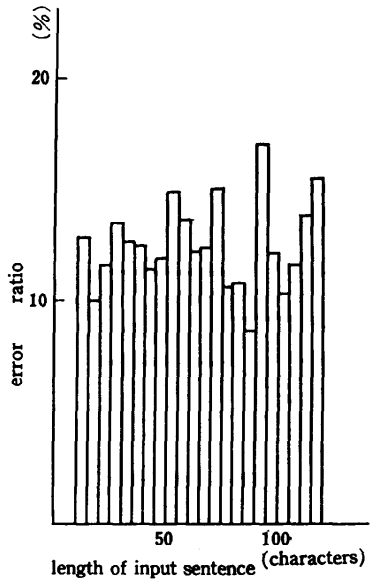


図-3 最初に出力された解析の誤り率（最長一致法）

語，形式名詞，補助用言が登録されている．図-1に自立語辞書の検索回数の比較を示したのは，ディスクパック上におかれた自立語辞書の検索は時間がかかるため，自立語辞書の検索回数が処理時間にほぼ比例し，処理時間の比較に使えるからである．

ヒューリスティクスを導入する場合に注意すべきことは，処理能率が大幅に落ちないようにしなければならないことである．最長一致法や文節数最小法は処理能率をほとんど落とすことなく最適解を出力することができ，最近のワードプロセッサの多くは文節数最小法を用いているようである．

4. べた書き言語のための機械辞書

日本語は単語の正書法が緩やかで，漢字，漢字かな混り，かな書き，送り仮名の省略などの単語表記が可能であり，機械辞書の見出し語数は実質的な単語数よりもかなり多くなる．

例 繰返す
　繰り返す
　繰りかえす
　くりかえす

日本語のもう一つの特徴は，単語単位に分かち書きされる欧米語などと異なり，べた書きされることである．したがって，入力文字列のどの位置からどの位置までが単語であるか，にわかには判断できず，機械辞書の検索回数が非常に多くなる傾向がある．一般に，機械辞書はディスクパックなどの二次記憶上にあるので，辞書の検索には時間がかかる．このようなことから，辞書の検索回数を極力少なくする工夫が必要となる．**3.2**で述べたように，形態素解析では入力文字列上の任意の文字位置から文末の方向に横たわるすべての単語を検出する必要がある（**3.2** の $WS(i)$）．

例 …くるまだいそげと…

苦
来る
車
車代

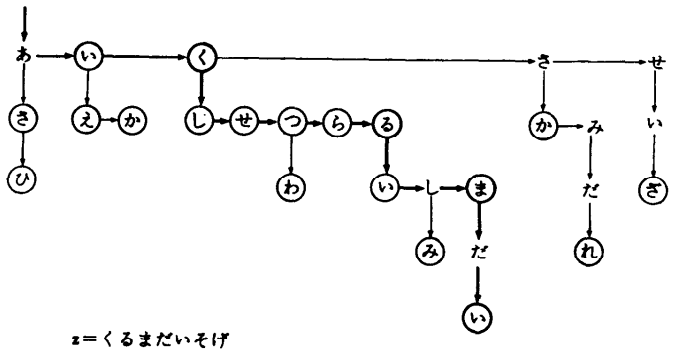
任意に与えられた文字列の左端に位置するすべての単語を，1回の検索で抽出するデータ構造には TRIE と拡張 B-tree がある．

4.1 TRIE

TRIE は R. Briandais によって提案された二進木で，**図-4** の見出し語集合に対して**図-5** の二進木を構成する．TRIE の各ノードは左（図では垂直方向）ポインタと右ポインタをもち，そのノードで終わる見出し語があれば，その語の文法情報（品詞，活用の種類など）の格納場所へのポインタをもつ．図-5 では，内容ポインタをもつノードは円で囲っている．「くるま

あさ	く	くつわ	くるま
あさひ	くし	くら	くるまだい
い	くせ	くる	さ
いえ	くち	くるい	さか
いか	くつ	くるしみ	さみだれ
			せいざ

図-4 見出し語



z＝くるまだいそげ

図-5 TRIE における，z の最左部分語検索

だいそげと…」の最左部分語検索は，まずルートノードから出発して，“く”のノードにたどりつくまで右ポインタを次々にたどる．“く”のノードにたどりつくと，左ポインタを1回たどり，二番目の文字“る”のノードにたどりつくまで右ポインタを次々にたどり，以後同様の動作を繰り返す．図-5 には辿る path が太線で示されている．このようにすると，path 上で左ポインタをたどるたびにその左ポインタを出しているノードの内容ポインタを参照することにより，“くるまだいそげ”のすべての最左部分語“く”，“くる”，“くるま”，“くるまだい”が検索できる．TRIE は見出し語の数が大きい辞書には利用できないが，付属語辞書などの見出し語数が小さな辞書には好都合である．

4.2 拡張 B-tree

見出し語数の大きい機械辞書では，一般に B-tree が用いられる．図-6 に図-4 の見出し語をもつ B-tree を示す．ここで，各ブロックは二次記憶から主記憶へのデータ転送の単位であり，カタカナ文字列は索引キーである．図-6 の B-tree 上で“くるまだいそげ”のすべての最左部分語を検索するためには，“く”，“くる”，“くるま”，…，“くるまだいそ”，“くるまだいそげ”を検索キーとして，計7回検索する必要がある．図-7 は，同じ見出し語を拡張 B-tree に収容したものである．拡張 B-tree では，葉（左端のブロック）以外のノード（ブロック）に索引キーとデータキー（見出し語）が混在する．“くるまだいそげ”の最左部分語の

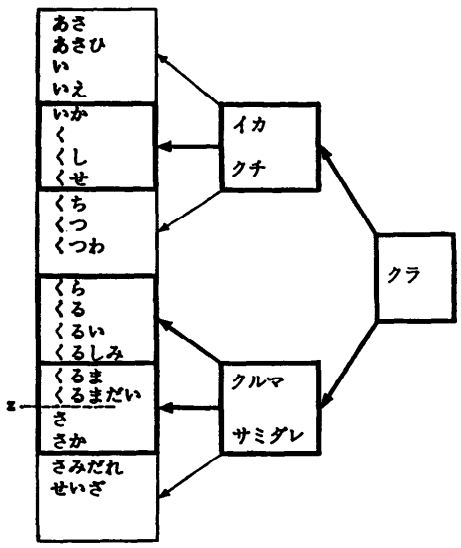


図-6 B-tree

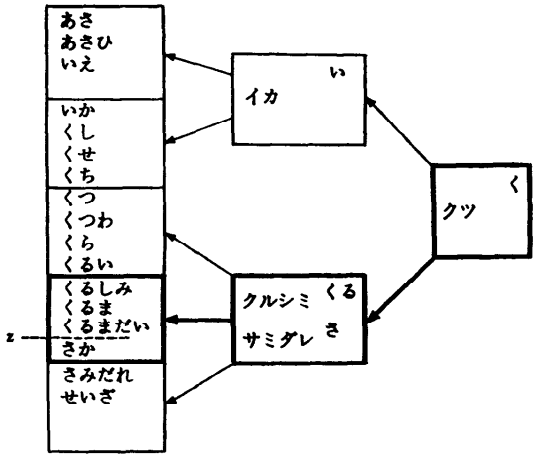


図-7 拡張 B-tree

検索は“くるまだいそげ”を検索キーとして，根（右端のブロック）から次々にポインタをたどり葉に到達する．根では“クツ”<“くるまだいそげ”だから，索引キー“クツ”の後のポインタにより次のブロックが選ばれる．次のブロックでは，“クルシミ”<“くるまだいそげ”<“サミダレ”だから，“クルシミ”と“サミダレ”の間のポインタにより葉のブロックが選ばれる．ポインタをたどる手順は B-tree の場合と同様である．ただ B-tree の場合と異なる点はポインタをたどってアクセスされたブロックに検索キー“くるまだいそげ”の最左部分語がすべてデータキーとして収容されていることである．図-7 の太線でたどる path が示されているが，この path 上のブロックに，検索キー“くるまだいそげ”のすべての最左部分語“く”，“くる”，“くるま”，“くるまだい”が収容されている．したがって，検索キーの最左部分語が1回の検索で，すべて検索できるしかけになっている．拡張 B-tree はブロック単位に，二次記憶上に分散配置できるので，大規模辞書に向いている．拡張 B-tree は詳細は文献2)を参照されたい．なお，本章の例題はかな見出しの例を用いたが，漢字見出しであっても本質的には同じ現象が発生するので，本章の例題は一般性を失っていない．

参 考 文 献

1) Sloat, C., Taylor, S. H. and Hoard, J.: Introduction to Phonology, Prentice-Hall, Englewood Cliffs, NJ (1978).
2) 日高，稲永，吉田：拡張 B-tree と日本語単語辞書への応用，電子通信学会論文誌 (D)，Vol.

J 67-D, No. 4, pp. 399-404 (1984).
3) Koskenniemi, K.: A General Computational Model for Word-Form Recognition and Production, Proc. of COLING '84, pp. 178-181 (1984).
4) Shieber, S.: The Design of a Computer Language for Linguistic Information, Proc. of COLING '84, pp. 362-366 (1984).
5) 吉村，日高，吉田：文節数最小法を用いたべた書き日本語文の形態素解析，情報処理学会論文誌，Vol. 24, No. 1, pp. 40-46 (1983).
6) 村木新次郎：動詞形態範疇の構造，特定研究言語情報処理の高度化研究報告 4，pp. 33-38 (1988).
7) 佐野，杉村，赤坂，久保：語形成に基づく形態素解析，情報処理学会自然言語研究会報告，NL 66-3, pp. 1-8 (1988).

(平成元年7月3日受付)