日本語文書処理　8－1
（1986・9・17）

日本語文の形態素解析アルゴリズム

吉村　賢治　　渡辺　美津乃　　津田　健蔵　　首藤公昭

福岡大学　　工学部

本稿では，表の広がりを多段階に制御する表方式の形態素解析アルゴリズムとその実験について報告する．このアルゴリズムでは，最尤候補を決定するためのヒューリスティックとしてコスト最小法を用いており，表の広がりを制御するために文節末の可能性の強さに関するヒューリスティックを用いている．解析実験は，コスト最小法だけを用いた表方式のアルゴリズムと，それに表の広がりを制御する機構を取入れたアルゴリズムについて行い，解析の精度と能率に関して比較した．その結果，表の広がりを制御するアルゴリズムは制御しないものに比べて解析の精度は悪くなるが，解析に要する時間は１／３程度になることが確認できた．

An Algorithm for Morphological Analysis of Japanese Sentences

Kenji YOSHIMURA, Mitsuno WATANABE, Kenzo TSUDA and Kosho SHUDO

Department of Electronic Engineering, Fukuoka University,
8-19-1 Nanakuma Jonan-ku, Fukuoka 814-01, Japan

An algorithm for morphological analysis of Japanese sentences is described. The algorithm is basically a tabular method, but has a mechanism to control the growth of the parse table. We employ two kinds of heuristic in this algorithm. One is a heuristic to select the correct structures named The Least Cost Method, and the other is a heuristic on a possibility of the ending of BUNSETSU (a sentence unit of Japanese) to control the growth of the parse table. By the experiment, we compared the accuracy and the efficiency of the algorithm that uses only The Least Cost Method with those of the algorithm that uses both heuristics. The accuracy of the latter is a little worse than the former's, but the latter seems to require only one third processing time of the former to analyze sentneces.

1．まえがき

実用的な日本語文解析システムにおいて，入力文中に存在する未登録語の位置や品詞等の推定は不可欠な処理である．筆者らは，参考文献(1) で入力文中の未登録語に対処できる表方式の形態素解析アルゴリズムについて報告した．このアルゴリズムは，文節数最小法(2) の考え方を一般化したコスト最小法の枠組みの中で，単語辞書に登録されていない片仮名・アルファベット列や一文字の漢字，平仮名を単語と同等に扱うことで未登録語に対処しており，約 95%の精度で未登録語の品詞を推定することができる(3)．しかし，一文字の漢字や平仮名を単語と同等に扱っているために入力文の一文字毎に自立語辞書の検索を行うという欠点を持っている．この欠点を補うものとして，筆者らは文節末の可能性の強さに関するヒューリスティックを用いて解析表の展開を多段階に制御するアルゴリズムを提案した(4)．本稿では，コスト最小法における各コスト値の設定と表の広がりを制御するアルゴリズムを示し，表の広がりを制御するアルゴリズムと制御しないアルゴリズムに対して行った解析の精度と能率に関する比較実験の結果について報告する．

2．文法モデル

2．1　日本語文の構造

アルゴリズムの記述を簡潔に行うために，ここでは次のように簡略化した日本語文の構造に基づいて議論を行う．

(1) 〈文〉::=〈文節〉|〈文〉・〈文節〉
(2) 〈文節〉::=〈自立語〉|〈文節〉〈付属語〉

本稿で述べるアルゴリズムでは，(2) で示される文節の構造を規定する規則のみを用いて，(1) に示されている文の構造を規定する規則は用いない．
なお，筆者らの形態素解析システムでは〈付属語〉として一般の付属語の概念を拡張した関係表現と助述表現と呼ぶ付属語的表現を用いているが，本稿の議論においてこれらの区別は重要ではないため，以下では単に付属語と呼ぶ．

2．2　文節の文法モデル

本節では 2.1の(2)で示した文節の構造を形式的に定義する．以下の議論では長さ nの文字列を sで表し，先頭から i番目の文字をs(i)で表す．つまり，s=s(1)s(2)…s(n)とする．

［定義１］　諸述語の定義
(1) 入力文字列の部分列s(i+1)s(i+2)…s(j)が文法情報αの単語であることを word(i,j,α)で表す．
(2) αが自立語の文法情報であることを cword(α)で表す．
(3) αが付属語の文法情報であることを fword(α)で表す．
(4) 文法情報αの単語W_1と文法情報βの単語W_2の接続W_1W_2が文節内で可能であることを connect(α,β)で表す．
(5) 文法情報αをもつ単語で文節を終われることを end(α)で表す．■

ここで，文法情報とは品詞，活用型，活用形などの情報である．
これらの述語を用いて，(2) で示される文節の構造を規定する規則は次のように表現される．

［定義２］　文節の文法モデル
文字列 s=s(1)s(2)…s(n) に対して，整数 $i_0, i_1, \cdots, i_{m+1}$が存在し，　($i_0=0, i_{m+1}=n, m\geqq 0$)
(1) word(i_j, i_{j+1}, α_j)　($j=0,1,\cdots,m$)
(2) cword(α_0)
(3) connect(α_j, α_{j+1})　($j=0,1,\cdots,m-1$)
(4) end(α_m)
を満たすとき文字列sは文節であると定義する．ただし，m=0のとき(3)は除外する．■

この文節の文法モデルは，文節を構成する単語の並びを隣接する２単語間の接続ルールだけで規定しているため，文節であるための必要条件にしかすぎないが，正しい日本語文を解析する場合にはこのモデルで十分である．

3．形態素解析アルゴリズム

3．1　アルゴリズム記述のための準備

本章では，２章で定義した文節の文法モデルに基づいて日本語文の形態素解析を行うアルゴリズムについて述べる．このアルゴリズムは基本的には表方式のアルゴリズムであり，入力文を文頭から文末に向かって走査して解析表を作成するアルゴリズムと，この解析表を文末側から文頭側に走査して解析結果を抽出するアルゴリズムとで構成される．
以下，アルゴリズムを簡潔に記述するために幾つかの定義を行う．

［定義３］　集合 Cset, Fset

集合 Cset(i) と Fset(i) を次のように定義する．

Cset(i)={(i,j,α) | word(i,j,α)&cword(α)}

Fset(i)={(i,j,α) | word(i,j,α)&fword(α)}

■

入力文字列sを与えてCset(i)を求めることは，入力文字列sの部分列s(i+1)s(i+2)…s(n)の先頭から始る自立語をすべて求めることに対応し，Fset(i)を求めることは，入力文字列sの部分列s(i+1)s(i+2)…s(n)の先頭から始る付属語をすべて求めることに対応する．ここでは単語辞書の詳細なデータ構造については触れないが，与えられた文字列の先頭から始まるすべての単語を，一定の時間以内で検索できる単語辞書に適したデータ構造として，単語を構成しているそれぞれの文字をノードとした木構造であるTRIE構造[5]がある．しかし，TRIE構造の一つのノードは，文字を格納する場所と他のノードを指す二つのポインタと辞書内容を指すポインタのための場所を必要とするため，単語辞書の容量が大きくなるという欠点をもつ．筆者らのシステムでは，TRIE構造を改良して単語辞書の容量を小さくする工夫をしている（図1）．

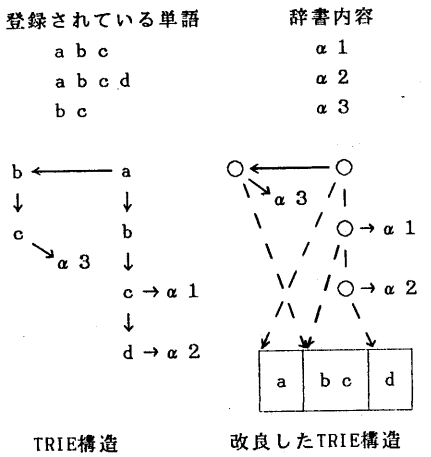

図1　単語辞書のデータ構造

3．2　ヒューリスティック

本稿で報告するアルゴリズムでは二つのヒューリスティックを利用している．

形態素解析において，複合語の分割や未登録語の処理を行うためには2章で述べた文節の文法モデルだけでは能力が弱く，この文法モデルを満足する解析結果の中には誤った解析も多数含まれている．これらの中から正しい解析を効率良く見つけるために，本アルゴリズムでは文節数最小法の考え方を一般化したコスト最小法を用いている．文節数最小法は自立語のコストを1，付属語のコストを0とした場合のコスト最小法と考えることができる．文法情報αに対して，そのコストを与える関数をcost(α)とする．ここで関数 cost の値は非負整数とする．どのような分類に対して，いくらの値を与えるかなどの具体的なことについては，4章で示す．

まえがきで述べたようにコスト最小法だけを用いた表方式のアルゴリズムでは，未登録語に対処するために片仮名・アルファベット列や一文字の漢字，平仮名を単語と同等に扱うために入力文の一文字毎に自立語辞書の検索が行われる可能性がある．それを避けるために本アルゴリズムでは，文節末の可能性の強さを用いて解析表の広がりを制御している．このために，文節末の可能性の強さを表す関数bendを述語 end を用いて定義する．この定義では，文節末の可能性の強さをδ段階に分類しているものとし，値が大きいほど文節末の可能性が強いとする．

［定義4］　関数 bend

文法情報αに対して，

もし end(α) ならば bend(α)=0，

そうでないならば，文法情報αおよびその他の条件（入力文における字種情報など）に応じて，

bend(α)=1,2,…,δ

とする．■

関数 bend の具体的な定義についても4章で示す．

3．3　アルゴリズム

入力文字列における位置を表す整数 i，j，文法情報α，文節末の可能性の強さを表す値 e，コストの部分和 w，状態を表す整数 c（c=0 または c=1）からなる6項組　(i,j,α,e,w,c) を項目と呼び，項目を要素とする集合 T(0),T(1),…,T(n) の全体を解析表と呼ぶ．また，c=0 である項目を非活動中の項目と呼び，c=1 である項目を活動中の項目と呼ぶ．文節数最小法やコスト最小法では，解析表を作成する時点で不要な項目は捨てていたが，このアルゴリズムでは，解析表を狭く展開する時点で不要であると判断した項目でも，後戻りが起こって，より広い解析表を展開する時点で必要になる可能性がある．そのときに単語辞書を重複して検索することを避けるために，不要な項目も捨てずに非活動中の状態にして登録する．

解析表の作成は，入力文字列を文頭側から文末側に向かって走査する過程で，単語に対応する項目を作り，適当な集合 T に登録することにより行われる．この過程で，作成された c=1 である項目にお

ける j の最大値を変数 Rmax で記憶する．

Ctable(0),Ctable(1),…,Ctable(n-1),
Ftable(1),Ftable(2),…,Ftable(n-1)

は共に値として０か１をとる配列で，Ctable(i)の値は入力文字列における位置 i で自立語辞書の検索を行ったか否かを，Ftable(i) の値は同じく付属語辞書の検索を行ったか否かを示している．
また，

Etable(1),Etable(2),…,Etable(n)

は値として 0,1,…,δ をとる配列で，Etable(i)の値は入力文字列の位置 i における文節末の可能性の強さの最大値を示している．本稿のアルゴリズムでは， Etable の値に従って解析表の広がりをδ段階に制御する．解析表を作成している過程でどの段階にあるかを変数 Phase で表す．

次にアルゴリズムの記述に用いる幾つかの基本的な手続きを定義する．

解析表に項目を登録する手続き tadd((i,j,α,e,w,c)) を次のように定義する．

［手続き tadd］
与えられた項目 (i,j,α,e,w,c) に対して，
c = 1 ならば，
　項目 (i,j,α,e,w,c) を T(j) に登録する．
　j > Rmax ならば Rmax ← j.
　e > Etable(j) ならば Etable(j) ← e.
　e > Phase ならば Phase ← e.
c = 0 ならば，
　項目 (i,j,α,e,w,c) を T(i) に登録する．

■

逆に T(i) から項目 (i,j,α,e,w,0) を削除する手続きを tdelete((i,j,α,e,w,0)) とする．

次に，入力文字列における位置 j から始る自立語の項目を作成して解析表に登録する手続きcmake(j) を定義する．

［手続き cmake］
Ctable(j) = 0 ならば，
　Cset(j) を求める．
　T(j) 中の e > Phase であるすべての項目 (i,j,β,e,v,1) における v の最小値を求め，それを w とする．
　Cset(j) のすべての要素 (j,k,α) について，
　　tadd((j,k,α,bend(α),w+cost(α),1)).
　Ctable(i) ← 1.

■

また，入力文字列における位置 j から始る付属語の項目を作成して解析表に登録する手続きfmake(j) を次のように定義する．

［手続き fmake］
Ftable(j) = 0 ならば，
　Fset(j) を求める．
　Fset(j) のすべての要素 (j,k,β) について，
　　T(j) に connect(α,β) を満たす項目 (i,j,α,e,v,1) が存在するならば，
　　そのような v の最小値 w について，
　　tadd((j,k,β,bend(β),
　　　cost(β)+w,1)).
　　存在しないならば，
　　tadd((j,k,β,bend(β),0,0)).
　Ftable(j) ← 1.
Ftable(j) = 1 ならば，
　T(j) の中のすべての要素 (j,k,β,e,0,0) について，
　　T(j) に connect(α,β) を満たす項目 (i,j,α,g,v,1) が存在するならば，
　　そのような v の最小値 w について，
　　tdelete((j,k,β,e,0,0)).
　　tadd((j,k,β,e,
　　　cost(β)+w,1)).

■

これらの手続きを用いて解析表を作成するアルゴリズムは次のようになる．

［解析表作成のアルゴリズム］
［０］　初期設定
　Rmax ← 0.
　Phase ← δ.
　Etable(i) ← 0.　　(i=1,2,…,n)
　Ctable(i) ← 0.　　(i=0,1,…,n-1)
　Ftable(i) ← 0.　　(i=1,2,…,n-1)
　T(i) ← ε.　　(i=0,1,…,n)
　p ← 1.
［１］　文頭の自立語の検索
　cmake(0).
［２］　位置 p からの単語の検索
　fmake(p).
　Etable(p) ≧ Phase ならば cmake(p).
［３］　ポインタの設定
　p ← p + 1.
　p = n ならば終了.
　p > Rmax ならば，
　　Phase ← Phase - 1.
　　Etable(P)=δ または P=1 になるまで，
　　　p ← p - 1.
　　［２］に行く．

■

項目が解析表に属することについては，次の定理が成り立つ．

［定理１］　項目の意味

解析表作成のアルゴリズムを実行している過程でT(k) に項目 (i,j,α,bend(α),w,c) が存在することは次のことを意味する．

(1) c = 0 のとき．

i = k かつ w = 0 であり，入力文字列の部分列s(i+1)s(i+2)…s(j) は文法情報αをもつ単語であるが，文節の文法モデルを満足する文頭からの単語の連鎖が解析表中に存在していない．

(2) c = 1 のとき．

j = k かつ，入力文字列の部分列s(i+1)s(i+2)…s(j) は文法情報αをもつ単語であり，文節の文法モデルを満足する文頭からの単語の連鎖が解析表中に存在し，そのような連鎖のコストの総和の最小値は w である．■

証明はアルゴリズムより明らかであるので省略する．

アルゴリズムにおいて，条件 p > Rmax が成り立ったときの処理が後戻りの処理である．ここでは，二つ以上前にある文節末の可能性が最も強い位置（Etable(p) = δになる位置）まで後戻りしても解析の精度が改善されないという実験結果から，一つ前にある文節末の可能性が最も強い位置まで後戻りしている．

このアルゴリズムを実行した結果 T(n) に状態が１で文節末の可能性が０でない項目が存在するならば，入力文の形態素解析の結果を出力することができる．本アルゴリズムは未登録語に対処しているために，このような項目は必ず存在する．解析結果を抽出するためには，まず上記の条件を満たす T(n) 中の項目でコストの総和（項目の第５項目）が最小であるものを選び，以後解析表を文頭側に向かって述語 connect を満たす，状態が１の項目を順次取り出せばよい．解析結果抽出のアルゴリズムは明らかなため，形式的な記述は省略する．

解析表作成のアルゴリズムの能率については次の定理が成り立つ．

［定理２］　解析表作成アルゴリズムの能率

解析表作成アルゴリズムの能率は，入力文字列の長さ n に対して時間計算量，領域計算量共にO(n)である．

［証明］　集合 T(j) に属する項目の個数を考える．集合 T(j) には次の２種類の項目が存在する．

(i,j,α,bend(α),w,1)

(j,k,β,bend(β),0,0)

単語の長さおよび一つの文字列がなり得る単語の個数（同型異義語の個数）は有限であるから，これら２種類の項目において j を固定した場合，i，αおよび k，βが取り得る値の個数はある定数でおさえることができる．また，一つの i，j，αの組合わせに対して w の値は一通りである．したがって，集合 T(j) に属する項目の個数はO(1)であるから，解析表全体の大きさはO(n)である．配列 Etable，Ctable，Ftable の大きさもそれぞれO(n)であるか

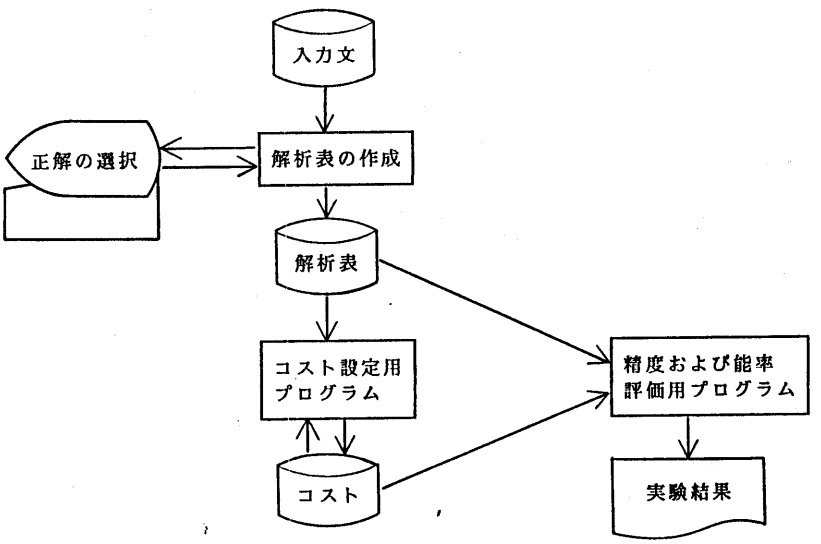

図２　実験の手順

ら，解析表作成アルゴリズムの領域計算量はO(n)である．

前半の議論と同様にして集合 Cset(j)，Fset(j)，T(j) に属する項目の個数はO(1)であるから，手続き cmake，fmake の一回の実行の時間計算量はO(1)である．解析表作成アルゴリズムにおいて，ステップ［2］，［3］の実行は高々δ × n 回である．ここでδは関数 bend の定義で決まる定数であるから，解析表作成アルゴリズムの時間計算量はO(n)である． ■

4．実験

4．1 実験の手順

実験の手順を図2に示す．まず，解析表作成プログラムで入力文を構成する可能性のあるすべての単語の項目を含む解析表を作成し，正しい解析を構成する項目に人手で印を付けてファイルに格納する．コスト設定用プログラム，解析の精度および能率の評価用プログラムは，この解析表のファイルを入力として処理を行う．実験システムは主記憶 2MB の VAX11/750 VAXVMS 上に FORTRAN で実現した．実験には約30,000語の自立語辞書と約4,000語の付属語辞書を用いた．

4．2 コストの設定

解析表を構成する項目のコストは，自立語，付属語，活用語尾，片仮名・アルファベット列，一文字漢字，一文字平仮名，記号（句読点）の単位で設定する．これらのコストの値は，すべてのコストの初期値を0としておき，繰り返し修正法を用いて決定した．それぞれのコストを C_i (i=1,2,…,7) とすると，一つのコストの修正方法は，

$C_i \leftarrow C_i + 1$

$C_i \leftarrow C_i$（修正なし）

if $C_i > 0$ then $C_i \leftarrow C_i - 1$

の3通りとし，一つの入力文に対して一つのコストを修正する．修正の対象とするコストは，修正後にコストの総和が最小となる解析結果（一般に複数個存在する）について，

(1) 誤り率の最小値が最小になるもの，

を選択する．このようなコストが一意に決らない場合には，コストが一意に決るまで以下の条件を順に調べる．

(2) 解析結果の個数が最小になるもの，

(3) 誤り率の最大値が最小になるもの，

(4) コストの修正をしないもの，

(5) コストを減らす方向に修正するもの．

ここで，誤り率とは，

$$\frac{\text{解析を誤った部分の入力文における長さ}}{\text{入力文の長さ}}$$

である．また，繰り返し修正の過程で，コストの総和が最小となる解析結果の個数が 20,000個以上になる入力文は評価の対象から除外した．

コスト設定用のデータとしては，科学技術雑誌の一論文を構成する 662個の句読点で終る文字列（総文字数 14,275）を用いた．このデータを繰り返し用いてコストの繰り返し修正を行ったときのコストの変化を図3に示す．グラフの横軸は 662文に対する繰り返しの回数，縦軸は 662文に対する繰り返し修正が終った時点でのコスト値である．ここで記号

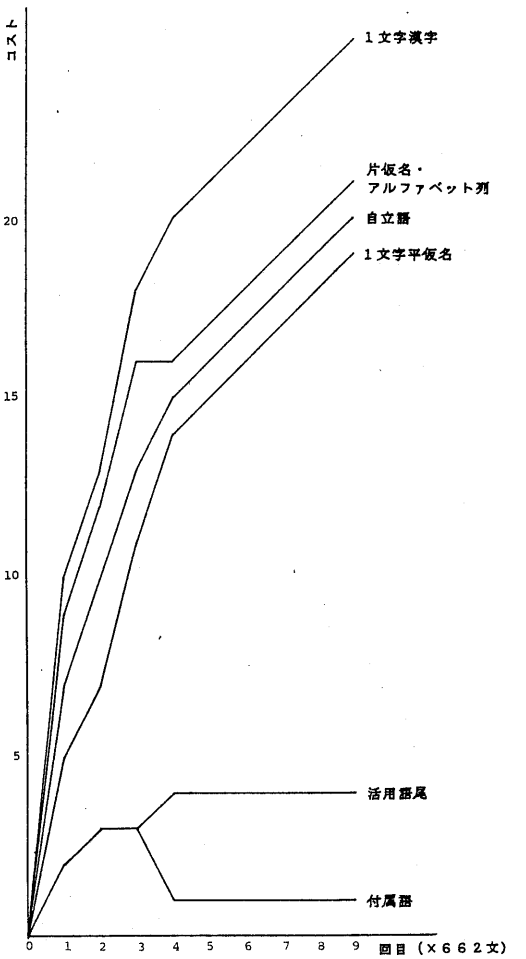

図3 コストの変化

のコストは他に競合するものがないので０に固定している．図３において，４回目で付属語と活用語尾のコストは収束しているが，その他のコストはそれ以降１ずつ増加していき，収束する様子はない．しかし，４回目以降，662文に対する誤り率の平均値やコストの総和が最小となる解析結果の個数の平均値は変化していないため，４回目で収束したものとみなした．このときの各コストの値は次のようになる．

自立語：	１５
付属語：	１
活用語尾：	４
片仮名・アルファベット列：	１６
一文字漢字：	２０
一文字平仮名：	１４
記号：	０

このコストのとき，コストの総和が最小となる解析結果における誤り率の最小値の 658文（4文を除外）に対する平均は 4.76% で，コストの総和が最小となる解析結果の個数の平均は，約 114個である．また，このコストの周辺のコストを調べた結果，付属語のコストを０にしたとき， 655文（7文を除外）に対する誤り率の最小値の平均が 4.15% になることが分かったが，コストの総和が最小となる解析結果の個数の平均が約 251個となるため，ここでは上記のコストを採用することにした．

４．３　文節末の可能性の強さ

解析表の広がりの制御に用いる，文節末の可能性の強さは表１に示す８段階に分類する．したがって，表の広がりは７段階に制御されることになる．表１において字種変化とは，注目している単語の末尾の文字が平仮名で，その文末側の単語の先頭の文字が非平仮名であることを示している．また，自立語，付属語，活用語尾における文節末の可能性の強弱は品詞，活用形で決定している．

表１　文節末の可能性の強さ

	文節末	字種変化	強さ
自立語，付属語，活用語尾	可（強）	有	7
	可（強）	無	6
	可（弱）	有	5
	可（弱）	無	4
	不可	－	0
片仮名・アルファベット列	－	－	3
一文字漢字	－	－	2
一文字平仮名	－	－	1

４．４　解析精度と能率の比較

表の広がりを制御するアルゴリズムと制御しないアルゴリズムに対する，解析精度と能率の比較実験は，コストの設定に用いた 662個の文字列と，同じく科学技術雑誌から取り出した別の一論文を構成する 692個の句読点で終る文字列（総文字数15,587）を用いて行った．前者をテキスト１，後者をテキスト２とよび，表の広がりを制御しない方式を旧方式，制御する方式を新方式とよぶ．実験の結果を表２に示す．

表２において， Emin，Emax の欄はコストの総和が最小となる解析結果における誤り率の最大値と最小値の平均で，解析結果の個数の欄は，同じくコストの総和が最小となる解析結果の個数の平均を表している．また，自立語辞書の検索回数および付属語辞書の検索回数の欄は，一つの文字列の解析中に行った自立語辞書および付属語辞書の検索回数の平均を表している．

なお，コストの総和が最小となる解析結果の個数が 20,000個以上になって評価の対象から除外した文字列の個数は，両方式ともにテキスト１では４個，テキスト２では１個であった．コストの総和が最小となる解析結果のあいまいさが多いが，その原因は今回実験に用いた付属語辞書における単語のカテゴリーが構文的な情報だけでなく意味的な情報も区別して分類されており，その数も計数しているためである．例えば，多くの意味を取り得る助詞の一つである格助詞「に」に対しては常に４通りのあいまいさが生じている．

また，コストの総和が最小となる解析結果の中に正しい解析（誤り率が０の解析）を含んでいたものは，テキスト１に対して旧方式では 467個，新方式では 425個，テキスト２に対して旧方式では 459個，新方式では 414個であった．

４．５　実験結果の考察

コスト最小法において解析の精度を悪くしている原因は，付属語辞書の不備（＜動詞・連用形＞＋かねない）や長単位の付属語的表現が本質的にもつあいまいさ（を用いている）である．表の広がりを制御するアルゴリズムの解析精度を改善するためには，

表2　実験結果

テキスト1

方式	Emin (%)	Emax(%)	解析結果の個数（個／文）	自立語辞書の検索回数（回／文）	付属語辞書の検索回数（回／文）
旧方式	4．71	20．0	114	27．2	25．5
新方式	5．27	20．2	122	8．20	19．6

テキスト2

方式	Emin (%)	Emax(%)	解析結果の個数（個／文）	自立語辞書の検索回数（回／文）	付属語辞書の検索回数（回／文）
旧方式	5．04	20．0	159	28．3	26．9
新方式	5．98	20．7	159	8．87	20．6

まず，表の広がりを制御しないコスト最小法の解析精度を改善する必要がある．

表2からも明らかなように，表の広がりを制御するアルゴリズムは能率の面では改善されているが，解析の精度は広がりを制御しないものに比べて悪くなっている．その主な原因としては，平仮名書きの自立語が存在して，狭く展開した表の中に正解が入らないということが挙げられる．このためには，頻繁に平仮名書きされる自立語の辞書を用意して，付属語辞書の検索を行うときにこの辞書も検索する必要がある．

5．あとがき

解析表の広がりを制御する表方式の形態素解析アルゴリズムとその精度および能率に関する実験について述べた．今後の課題としては，

(1) 解析精度の改善
(2) 接辞処理の強化
(3) 数詞処理の強化

等がある．

最後に，本実験で使用した自立語辞書は「九州芸工大自立語辞書 KID-82」(6)の規模を9万語から3万語に筆者らが縮小したものである．KID-82 の使用を認めて頂いた九州芸工大・稲永紘之講師に感謝の意を表する．

参考文献

(1) 吉村・日高・吉田：未登録語を含む日本語文の形態素解析アルゴリズム，九州大学工学集報，Vol. 55，No. 6，1982.
(2) 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. 23，No. 6，1982.
(3) 吉村・日高・吉田：漢字仮名混り文の形態素解析とその実験，情報処理学会第28回全国大会講演論文集，5M-6，1984.
(4) 吉村・渡辺・津田・首藤：広がりを制御できる表方式の形態素解析アルゴリズム，情報処理学会第32回全国大会講演論文集，2T-5，1986.
(5) Knuth, D. E.: The Art of Computer Programming, Vol. 3, Sorting and Searching, Addison-Wesley Pub. Co., 1973.
(6) 稲永・吉田：日本語処理のための機械辞書，情報処理，Vol. 23，No. 2，1982.