知識工学と人工知能　59－10
（１９８８．７．７）

音韻法則を用いる共通語から琉球方言への単語変換システム

高良富夫*　新垣　薫*　志喜屋章†

*琉球大学工学部　†アドバンテック（株）

琉球方言は、現代では本土方言とかなり異なっている。しかし、言語学的研究によれば、琉球方言と本土方言は祖語を同じくすると言われる。両方言が同一の祖語から発したとされるひとつの根拠は、両方言の間に整然たる音韻の対応（音韻法則）が存在することである。

ここでは、この音韻法則の有効性をより客観的に検討するため、音韻法則をＯＰＳ５のＩＦ－ＴＨＥＮルールとして記述し、東京方言の単語から琉球首里方言の単語を生成する単語変換システムを構成した。このシステムで用いた音韻法則を東京方言の名詞に適用し、生成した単語を実際の琉球首里方言の単語と比較した結果、基礎語いの場合、50％が一致することがわかった。本システムでは、音韻法則を明確に規定することができるので、これまでの方言学のような人手に頼る研究に比較して、より客観的に音韻法則を検討することができる。

Word Transformation System for Translating Standard Japanese into the Ryukyu Dialect Using the Phonetic Law

Tomio TAKARA*, Kaoru ARAKAKI* and Akira SHIKIYA†

*Faculty of Engineering, University of the Ryukyus,
1, Senbaru, Nishihara, Okinawa, 903-01 Japan,
† Advantech Corp.

According to linguistic research, the mainland-Japanese dialects and the Ryukyu dialects have the same origin even though they have a very different form now. One factor which supports the thesis that they have the same origin is that a systematic law of phonetic correspondence exists between them.

In order to investigate the phonetic law more objectively, we propose a word transformation system which synthesizes words of the Ryukyu-Shuri dialect from that of the Tokyo dialect (standard Japanese). In the system, the phonetic law is embodied in the IF-THEN rules of the OPS5. As a result of applying the phonetic law used in the system to nouns of the Tokyo dialect and comparing the synthesized words with real nouns of the Ryukyu-Shuri dialect, it was found that there was a 50 % agreement between their basic vocabularies. Because the phonetic law can be clearly prescribed in the system, we can study the phonetic law more objectively than by the manual research of traditional dialectology.

1．まえがき

日本語は方言学上、まず本土方言と琉球方言とに大分類される。琉球方言とは、南西諸島で話される方言の総称である。琉球方言は現代では本土方言とかなり異なっており、琉球方言に属するどの方言も本土方言のどの方言ともまったく通じないほどである。しかし、言語学的研究によれば、琉球方言と本土方言とは同一の祖語から発したと考えられている(1)。

両方言が祖語を同じくすることの一つの根拠は、両方言間に存在する整然たる音韻の対応である。例えば、東京方言の／ame／（雨），／kumo／（雲）は、それぞれ琉球首里方言では／ami／，／kumu／であり、東京方言の母音／e,o／は、それぞれ琉球方言の／i,u／に対応する。このような音韻対応法則はいくつかあり、これらの法則を適用すれば、琉球方言の多くの単語は本土方言の単語に対応づけることができるといわれる。

音韻対応法則に関しては、言語学的によく分析されている(1)。しかし、これまでの研究方法は、研究者の方言語彙にたよる手作業的方法によるものであり、音韻対応法則の有効性を大量の語彙を用いて検証しようとする場合など、思わぬ見落しや、法則の矛盾した適用などが生じる恐れがあった。

そこで我々は、この音韻法則の有効性をより客観的に検討することを目的として、音韻対応法則を用いて共通語の単語から琉球方言の単語を生成する単語変換システムを作成した。このシステムは、エキスパートシステム構築用ツールであるＯＰＳ５を用いて構成した。音韻対応法則は「もし本土方言の音韻がＡである場合、琉球方言の音韻はＢである」という形式で書けるので、ＯＰＳ５のＩＦ－ＴＨＥＮルールの構文が有効に利用される。

ところで、琉球方言話者の多くは、音韻法則を無意識に適用していると思われる。すなわち、元来琉球方言の語彙に属さない単語は、音韻法則が適用され琉球方言らしく発音されることがある。例えば、人名／tomio／（富夫）は／tumioo／と発音されることがある。これは、前述の母音の対応法則が一部適用されている。このことから、本システムは、方言話者を模擬する方言話者エキスパートシステムと考えることができる。もちろん本システムは、言語学者の見い出した音韻法則を知識ベースとして持っていることから、方言学エキスパートシステムである。なお本システムは、現在のところ名詞だけを変換するものである。

次章以下では、まず琉球方言と本土方言の音韻対応について述べ、ＯＰＳ５について概観する。次に、単語変換システムについて具体的に述べ、最後にシステムの実行結果を示し、音韻法則の有効性について考察する。

2．琉球首里方言と東京方言の音韻対応

ここでは、琉球方言の中から、かつて琉球王府のあった首里の方言をとりあげる。これは、現代日本語における東京方言に相当し、琉球方言の共通語と考えることができる。また、本論文で単に共通語（または標準語）というときは概ね東京方言のことである。

首里方言と標準語との音韻対応に関しては文献（１）の解説篇を参照した。文献（１）では、首里方言で使用される音素のそれぞれについて、標準語の音素との対応関係が述べられている。そこでは、「～に対応する」「～に対応するのが普通である」「しばしば対応する」「多く～に対応する」と確定的な表現や、「ただし～」「対応する例が見られる」「～ことがある」などの例外的な表現で述べられているものがある。ここでは、確定的な記述だけを採用することとし、対応法則を表１のように整理した。表１では、対応するモーラの数で分類してあり、「共通語の音韻Ａは、首里方言の音韻Ｂに」対応するという形式で表現した。表１に現れていない音素は、首里方言と東京方言とで一致すると考えてよい。

表１の母音の前の記号”＃”は、共通語において子音の無いモーラであることを示し、記号”？”，”’”は、それぞれ首里方言の声門破裂子音および、その対立音(2)を示す。但し、今回のシステムでは、使用したソフトウエアにおいて文字”？”が使用できないため、これらの音素の差異は考慮されていない。

また、表１を作成するに当たって、文献（１）の表現をできるだけ忠実に表現することを基本としたが、プログラミングが容易となるように規則を整理した部分もある。表１から、規則を組み合わせれば、より少ない規則に整理することができると考えられるが、これについてはシステムの実現法と関連して今後の課題となっている。

表1　音韻対応表

1モーラが2モーラに
（1）#eは'iiに
（2）#aは?aaに
（3）#uは?uuに
（4）heとhiはhwiに
（5）kiとcuはciiに
（6）giとzuはziiに
（7）suはsiiに
（8）aはaaに
（9）iとeはiiに
（10）uとoはuuに

1モーラが1モーラに
（1）語頭の#aは?aに
（2）語頭の#uは?uに
（3）語頭の#eは'iに
（4）2モーラの語尾のmiはそのまま
（5）3モーラ以上の語尾のmiはNに
（6）heとhiはhwiに
（7）kiとcuはciに
（8）giとzuはziに
（9）suはsiに

2モーラが2モーラに
（1）語頭の#uiは?wiiに
（2）語尾のmonoはmuNに
（3）語中のariはaiに
（4）語中のuriとoriはuiに
（5）aiとaeはeeに
（6）aoはooに
（7）awaはaaに
（8）haiとhaeはhweeに
（9）kuraはQkwaに
（10）kureはQkwiに
（11）guraはNgwaに
（12）gureはNgwiに
（13）語中のuiとueはiiに

2モーラが1モーラに
（1）語頭のmiとmuは、次のモーラにi,uを含まないとき'Nに

子音が子音に
（1）語頭のjは'jに
（2）語頭のwは'wに
（3）語頭のrはdに
（4）rjはjに
（5）kjとcjはcに
（6）gjとzjはzに
（7）sjはsに

母音が母音に
（1）eはiに
（2）oはuに

後述するように、この音韻対応法則では説明できない単語も琉球首里方言には多数存在する。これらの単語は、本土方言の単語とは語源を異にすると考えられるが、その由来が不明であるものも少なくない。

3．ＯＰＳ５[3],[4]

本システムは、プロダクションシステムをベースにしたエキスパートシステム構築用ツールであるＯＰＳ５を用いて構成した。ＯＰＳ５は、プロダクション記憶、作業記憶、推論部の3要素から構成されている（図1）。プロダクション記憶は、ＩＦ－ＴＨＥＮ型ルールの集合で記述された知識ベースである。作業記憶は、対象の状況がルールのＩＦ（条件）部と同様の形式で表現された要素の集まりである。推論部は、実行可能なルールを、ある戦略（ＬＥＸ戦略）に基づいて選択、実行する機構である。

［作業記憶］

作業記憶は、作業記憶要素とタイム・ダグの対で構成されている。作業記憶要素のフォーマットは以下のようになっている。

（クラス名　^属性1　値1　^属性2　値2
・・・　^属性n　値n）

タイム・ダクは、その作業記憶要素がいつ作られたか、または修正されたかを表し、値が大きいほどその作業記憶要素は新しい。

［プロダクション記憶］

プロダクション記憶内のひとつのＩＦ－ＴＨＥＮ型ルールは、図2のようなフォーマットで表される。このルールの条件部と作業記憶要素が適合すると、動作部が実行可能になる。動作部を実行することによって、作業記憶の内容が変更されると、次のルールが適用される状態になる。動作部の動作命令としては、以下のようなものなどがある。

make　・・・新しい要素を作業記憶に加える。

modify・・・存在する作業記憶要素の値を変更する。

remove・・・作業記憶から要素を削除する。

［推論部］

推論部は、大きく分けて以下の4つのステップの動作を行う。

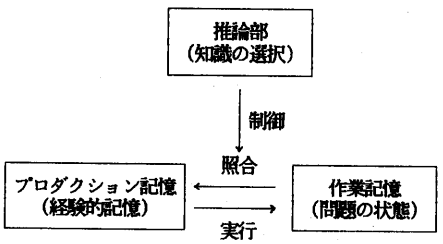

図1　ＯＰＳ５の構造

```
フォーマット
   (P  プロダクションルール名
         (条件要素1)
         (条件要素2)
           ⋮
         (条件要素n)
   →
         (動作1)
         (動作1)
           ⋮
         (動作1)              )
```

図2　プロダクションルール

ステップ1：照合

プロダクション記憶内にあるすべてのルールの条件部と作業記憶の現内容を照らし合わせ、条件部が満足されたルールを選び出す。この選び出されたルールのひとつひとつをインスタンシェーション、これらの集合を競合集合という。

ステップ2：競合解消

照合時に選び出された競合集合の中から指定された戦略に基づいて、ひとつのルールを選択する。

ステップ3：動作

競合解消によって選ばれたルールの動作部に示された動作を実行する。

ステップ4：繰り返し

ステップ1に戻る。

ステップ1～2は認識サイクル、ステップ3～4は行動サイクルとよばれる。「認識－行動」サイクルが繰り返されることによって推論が行われる。

ＯＰＳ５の推論部における競合解消に用いられるＬＥＸ戦略は以下のような競合解消法である。

（１）１度選択されたインスタンシェーションを競合集合から除く。（１つのルールが無限ループを起こすのを防ぐため）

（２）より新しい作業記憶要素を含むインスタンシェーションを選ぶ。ここで２つのインスタンシェーションを比較する場合、次の方法を用いる。

（ａ）最近のタイム・タグ（値の大きいもの）を比較し新しい方を選ぶ。

（ｂ）同じ場合、次に新しいタイム・タグを比較し、新しい法を選ぶ。

（ｃ）一方が選ばれるまで、あるいは、一方の作業記憶要素のタイム・タグがなくなるまで次々にタイム・タグを比較する。（残っている方が選ばれる。）同時に、作業記憶要素のタイム・タグがなくなった場合は、（３）へ進む。

（３）より条件の厳しいインスタンシェーションを選ぶ。つまり、プロダクションの条件部で行われる値のチェックの回数が多い方を選ぶ。

（４）以上で決まらなかった場合は、任意のインスタンシェーションが選ばれる。

本システムで用いたＯＰＳ５[4]は、Prolog-KABA上に実現されたものであり、ＩＦ－ＴＨＥＮルールの動作部は、Prologの述語により実行することもできる。また作業記憶要素の属性の値がベクトル量である場合、これをリストで表すことができる。ＩＦ－ＴＨＥＮルールの書式は、元来のＯＰＳ５と多少異なるが、その内容は容易に類推できる。

4．単語変換システム

4.1　概要

本システムは、共通語と琉球方言の間に存在する音韻法則を、エキスパートシステム構築用ツールであるＯＰＳ５上にＩＦ－ＴＨＥＮルールとして作成したものであり、これにより共通語の単語を琉球方言の単語へ変換する。今回構成したシステムでは、変換されるモーラ数に着目して、変換規則を表１のように分類し、それぞれをＩＦ－ＴＨＥＮルールで表現した。

共通語を琉球方言に変換するためにＯＰＳ５のＩＦ－ＴＨＥＮルールで必要な情報は、以下のものである。

入力段階における単語の

①文字集合（ひとつひとつの文字の集まり［リスト］）

②文字数（未変換の文字数；繰り返しの条件に使用される）

③モーラ数（ルールはその単語のモーラ数で異なる）

現在の変換タスクの対象である文字の

④位置

④の位置における入力情報である

⑤１文字目から最大５文字目まで

これらの情報は、文字列の入力の際または変換（変換は規則がないときの無変換も含む）の際、認識され、ＩＦ－ＴＨＥＮルールの条件部との照合、ルールの動作部の実行のために使用される。

本システムは、「入力段階」、「音韻変換段階」、

「出力段階」の３段階からなる。すなわち、本システムのすべてのＩＦ－ＴＨＥＮルールは、この３段階のいずれかに分類される。以下にそれぞれの段階について簡単に説明する。

［入力段階］

使用者がキーボードから入力した共通語のローマ字の文字列を”文字集合”に変換し、その”文字数”、”モーラ数”、注目する文字の”位置”、”１文字目”から”５文字目”までの文字を抽出し、次の段階に受け渡す。ただし、入力文字列が”end”である場合は終了する。

［音韻変換段階］

この段階は、本システムの本体であり、表１の６種のＩＦ－ＴＨＥＮルール群より成る。

ＩＦ－ＴＨＥＮルールの条件部では、入力段階から受け取った”文字集合”、”文字数”、”モーラ数”および文字の”位置”、”１文字目”から”５文字目”までの文字を認識する。

ＩＦ－ＴＨＥＮルールの動作部では、まずルールと一致した条件部の注目している文字を変換する。次に”文字数”から動作を行った文字の数を減じ、次の位置の５文字を読み込む。６種のどのＩＦ－ＴＨＥＮルールも適用されない場合は、単に、１文字だけシフトする（”文字数”から１だけ減じ、次の５文字を読み込む）。

音韻変換の終了条件は、”文字数”がゼロ（０）になった場合である。この時、変換した文字集合を出力段階へ受け渡す。

［出力段階］

変換された”文字集合”を文字列としてディスプレイに表示する。その後、作業記憶に残っている情報を消去し”位置”を１とし、タスクを入力段階へ受け渡し、上記の動作を繰り返す。

なお、入力段階、出力段階ともＯＰＳ５ではＩＦ－ＴＨＥＮルールで表現される。

本システムでは、”文字集合”をリスト形式で取り扱うため、Prologで記述されたリスト処理用の述語insertおよびdeleteが多用される。これらの機能は以下のとおりである。

insert(A,WW,X1,X2)・・・文字集合X1のＡ番目の位置に文字WWを挿入し、これを文字集合X2とする。

delete(A,X,X1)・・・文字集合ＸのＡ番目の文字を消去し、これを文字集合X1とする。

またProlog-KABA の組み込み述語以外の述語として他に、入力段階ではreadline、alter __mora、音韻変換段階でalter __moji5 、change1 、出力段階でputlist がある。

4.2　入力段階

（１）宣言文

本システムの先頭には以下のような宣言文がある。

```
1: literalize(入力,共通語).
2: literalize(入力段階,文字集合,文字数,モーラ数).
3: literalize(タスク,は,位置).
4: literalize(入力情報,１文字目,２文字目,３文字目,４文字
                目,５文字目).
```

これらの宣言文では、ルール中に使用される、作業記憶のクラス名およびそのクラスの属性名が宣言される。本システムで使用されるクラス名は、入力、入力段階、タスク、入力情報である。

また、本システムの最後には

```
make( タスク, ^は, 初期画面, ^位置, 1).
```

と記述されている。これは、本システムを起動した時の作業記憶の初期状態を与える（makeする）ものであり、これにより、最初のタスクを”初期画面”、注目する文字”位置”を”１”番目とする。

（２）初期画面

本システムのタイトルを表示する初期画面タスクである。これを終了するとタスクを”入力”とする。

（３）入力

このＩＦ－ＴＨＥＮルールは下のように記述されている。

```
13: p(入力,
14:     タスク(^は,入力),
15: -->,write(' 琉球方言に変換したい共通語を '),nl,
16:     write(' ローマ字で入力して下さい。'),nl,
17:     write(' ＊　終了の時はendと入力 '),nl,nl,
18:     readline(K),nl,
19:     make(入力,^共通語,K),
20:     modify(1,^は,入力チェック)).
```

これにより、まず変換したい共通語の単語をローマ字で入力するよう使用者に促す。（15～17行）。次に入力されたローマ字の文字列Ｋを読み込み（18行）、クラス名”入力”の属性”共通語”の値をＫとし（19行）、タスクを”入力チェック”とする。

なおreadlineは、ライン入力をするためのPrologで記述された述語である。

（４）入力チェック

入力が”end”のとき、タスクを”終了”とし、それ以外ではタスクを”入力認識”とする。

（５）入力認識

このＩＦ－ＴＨＥＮルールは下記の通りである。

```
31: p(入力認識,
32:     タスク(^は,入力認識),
33:     入力(^共通語,K),
34: —>,alter__mora(K,X,C,M),
35:     alter__moji5(1,X,W1,W2,W3,W4,W5),
36:     make(入力段階,^文字集合,X,^文字数,C,^モーラ数,
            M),
37:     make(入力情報,^１文字目,W1,^２文字目,W2,^３
            文字目,W3,
38:             ^４文字目,W4,^５文字目,W5),
39:     remove(2),
40:     modify(1,^は,変換規則)).
```

これにより、まず入力されたローマ字列”共通語”Ｋを認識し（33行）、”文字集合”Ｘに変換するとともに”文字数”Ｃ，”モーラ数”Ｍを抽出し（34行）、第”１文字目”から第”５文字目”までW1～W5を抽出する（35行）。抽出したそれぞれの値を新しい要素”入力段階”，”入力情報”として作業記憶に加える（36～38行）。次に”共通語”のローマ字文字列を削除し（39行）、タスクを”変換規則”とする（40行）。

なお、alter __mora（34行）およびalter __moji5（35行）は上述の機能を持つ述語であり、Prologで記述した。alter __moraは、モーラが「母音のみ」または「子音＋母音」または「子音＋子音＋母音」または「Ｎのみ」からなると仮定し、実行している。

4.3　音韻変換段階

4.3.1　１モーラが２モーラに

文献（１）によると、首里方言には１モーラの自立語はなく、共通語の１モーラの自立語はすべて２モーラとなる。例えば：

共通語	首里方言
／#e／（絵）	／'ii ／
／hi／（火）	／hwii／
／ke／（毛）	／kii ／

である。これらに関するルールは以下のように記述されている。

（１）母音のルール

このＩＦ－ＴＨＥＮルールは：

```
66: p(規則__moral__boin,
67:     タスク(^は,変換規則,^位置,A),
68:     入力段階(^文字集合,X,^モーラ数,1),
69:     入力情報(^１文字目,W,^２文字目,[]),
70: —>,change1(W,WW),
71:     delete(A,X,X1),
72:     insert(A,WW,X1,X2),
73:     modify(2,^文字集合,X2),
74:     modify(1,^は,出力)).
```

これにより、母音a,i,u はそれぞれaa,ii,uuに、ｅとｏはそれぞれii,uu に変換する（70行）。このルールは、１モーラの単語では例外（cu,zu,su）を除いてすべてに共通するので、”位置”は変数にした（67～69行，71～72行）。また１モーラ単語であるので、どの単語も母音の変換が終了すれば、すぐにタスクを出力段階へ受け渡す（74行）。なおchange1 （70行）は、上記の機能を持つ述語であり、Prologで記述した。

（２）１モーラのhiとheはhwiiに

このＩＦ－ＴＨＥＮルールは：

```
76: p(規則__moral__hi__he,
77:     タスク(^は,変換規則),
78:     入力段階(^文字集合,X,^モーラ数,1),
79:     入力情報(^１文字目,h,^２文字目,(i;e)),
80: —>,delete(1,X,X1),
81:     insert(1,hw,X1,X2),
82:     modify(3,^１文字目,i,^２文字目,[]),
83:     modify(2,^文字集合,X2),
84:     modify(1,^は,変換規則,^位置,2)).
```

これにより、１モーラのhiとheの場合（78～79行）、”文字集合”の１番目の文字ｈを削除し（80行）、文字hwを挿入し（81行）、”１文字目”と”２文字目”を書き換える（82行）。変換した文字集合X2は、あらためて属性”文字集合”に代入し（83行）、文字”位置”を２にし、タスクを母音のルールに受け渡す（84行）。

（３）１モーラのki,gi はcii,zii に

（２）と同様である。

（４）１モーラのcu,zu,suは各々cii,zii,sii に

子音はそのままにして、母音のｕをｉに書き換えてタスクを母音のルールに受け渡す。

（５）その他の１モーラ単語

上記の規則と一致しないもの（その他の子音＋母音、子音＋子音＋母音）は、それぞれの母音を第”１文字目”に、[]を第”２文字目”に代入して、母音のルールへタスクを受け渡す。

4.3.2　１モーラが１モーラに

このルールは、共通語の注目するモーラが１モーラであり、これを首里方言の１モーラに変換するルールの集合である。母音の変換については、後述する。

（１）２モーラの語尾のmiはそのまま

”モーラ数”が２、第”１文字目”がm、第”２文字目”がｉ、第”３文字目”が［］であることを認識すると、タスクを出力段階に受け渡す。

（２）３モーラ以上のmiはＮに

（１）と同様に認識し、ｍとｉを削除し、ｎを挿入する。タスクを出力段階に受け渡す。

（３）hiとheはhwi に

このＩＦ－ＴＨＥＮルールを下に示す。

```
153: p(規則_mora2_hi_he,
154:      タスク(^は,変換規則,^位置,A),
155:      入力段階(^文字集合,X,^文字数,C,^モーラ数,{M,
                    >=2} ),
156:      入力情報(^１文字目,h,^２文字目,(i;e)),
157: —>,delete(A,X,X1),insert(A,hw,X1,X2),
158:      B is A+1,
159:      delete(B,X2,X3),insert(B,i,X3,X4),
160       modify(2,^文字集合,X4,^文字数,compute(C-2)),
161:      modify(1,^は,次の規則,^位置,compute(A+2)),
162:      make(再認識)).
```

これにより、まず”モーラ数”が２モーラ以上であり、”１文字目”がｈ、２文字目がｉまたはｅであることを認識する（155 ～156 行）。hwの次の位置Ｂのｉまたはｅを削除し、ｉを挿入する（158 ～159 行）。変換を行った文字の数（２）だけ”文字数”から減じ（160 行）、文字の”位置”には２を加えるとともにタスクは”次の規則”にする（161 行）。

このとき”文字数”がゼロであるかを認識するため、またゼロでないときは変換した文字の次の位置から５文字を認識するため、”再認識”ルールへ移行する（161 ～162 行）。再認識については後述する。

（４）kiはciに、giはziに

（３）と同様である。

（５）cu,zu,suはそれぞれci,zi,siに

子音はそのままにし、母音のｕをｉに変換する。

4.3.3　２モーラが２モーラに

このルールは、注目するモーラが２モーラであり、これを２モーラに変換するルールの集合である。

（１）語尾のmonoはmuN に

このＩＦ－ＴＨＥＮルールを下に示す。

```
196: P(規則_t_mono,
197:      タスク(^は,変換規則,^位置,A),
198:      入力段階(^文字集合,X,^文字数,C,^モーラ数,{M,
                    >=2} ),
199:      入力情報(^１文字目,m,^２文字目,o,^３文字目,n,
200:                    ^４文字目,o,^５文字目,[] ),
201: —>,P is A+1,delete(P,X,X1),
202:      insert(P,u,X1,X2),
203:      Q is A+3,delete(Q,X2,X3),
204:      modify(2,^文字集合,X3),
205:      modify(1,^は,出力)).
```

これにより、条件部で語尾のmonoを認識すると（199 ～200 行）、”２文字目”のｏを削除し（201 行）、代わりにｕを挿入し（202 行）、”４文字目”のｏを削除する（203 行）。これを変換後の”文字集合”とし（204 行）、語尾の変換であったのでタスクを出力段階へ受け渡す（205 行）。

（２）語中のari はaiに

”２文字目”のｒを削除する。”再認識”へ。

（３）語中のuri,ori はuiに

”１文字目”を削除し、これをｕとし、”２文字目”を削除する。”再認識”へ。

（４）ai,ae はeeに

ai,ae を削除し、eeを挿入して”再認識”へ。

（５）aoはooに

（２），（３）と同様。

（６） awaはaaに

（２）と同様。

（７）hai,hae はhewee に

（８）kuraはQkwaに

（９）kureはQkwiに

(10)gura はNgwaに、gureはNgwiに

（７）～（10）は（４）と同様に作成した。

4.3.4　２モーラが１モーラに

このグループに属する音韻対応法則は、

（１）語頭のmiとmuは、次のモーラにｉ，ｕを含まないとき'Nに

だけである。これは共通語の注目するモーラが２モーラであり、変換されるモーラは１モーラである。このＩＦ－ＴＨＥＮルールは以下の通りである。

```
310: p(規則_hed_mi_mu,
311:      タスク(^は,変換規則,^位置,1),
312:      入力段階(^文字集合,X,^文字数,C,^モーラ数,{M,
                >=2} ),
313:      入力情報(^１文字目,m,^２文字目,(i;u),^３文字
                目,W3,  ^４文字目,(a;e;o)),
314: —>,delete(A,X,X1),
315:      delete(A,X1,X2),
316:      insert(A,n,X2,X3),
317:      modify(2,^文字集合,X3,^文字数,compute(C-2)),
318:      modify(1,^は,次の規則,^位置,compute(A+1)),
319:      make(再認識)).
```

これにより、まず語頭のmiまたはmuの次のモーラの母音がｉまたはｕでないことを認識する（313 行）。次にmiまたはmuを削除し（314 ～315 行）、そこにｎを挿入する（316 行）。”再認識”へ。

4.3.5　子音が子音に

子音だけに注目し、子音を変換するルール群である。

（１）語頭のｒはｄに

２モーラ以上の単語で、”位置”が１のとき、”１文字目”がｒである場合、ｒを削除し、ｄを挿入する。

（２）語中のrjはｊに

ｒを削除する。

（３）kjとcjはｃに

ｋまたはｃ、およびｊを削除し、ｃを挿入する。

（４）giとziはｚに

（３）と同様。

（５）sjはｓに

ｊを削除する。

これらのルールの適用後、いずれも”再認識”へ移行する。

4.3.6　母音が母音に

共通後の５つの母音は、短母音の場合、首里方言の３つの母音と次のように対応する。

従って、ｅをｉに、ｏをｕに変換する。”再認識”へ。

4.3.7　変換規則なし

4.3.1 から4.3.6 までのルールのいずれにも該当しない場合は、音韻を変換せず”文字数”を１だけ減じ、”位置”を１だけシフトして、”再認識”へ移行する。

4.3.8　再認識

これは、4.3.1 から4.3.7 までのルールを繰り返して適用することを可能にするための”再認識”のルールである。これは以下のように記述されている。

```
44: p(出力_条件,
45:      再認識,
46:      タスク(^は,次の規則),
47:      入力段階(^文字数,0),
48: —>,remove(1),
49:      modify(2,^は,出力)).
50:
51: p(ループ文,
52:      再認識,
53: —>,remove(1)).
54:
55: p(再認識,
56:      タスク(^は,次の規則,^位置,A),
57:      入力段階(^文字集合,X),
58:      入力情報(^１文字目,XX),
59: —>,alter_moji5(A,X,W1,W2,W3,W4,W5),
60:      modify(3,^１文字目,W1,^２文字目,W2,^３文字目,
                  W3,
61:                  ^４文字目,W4,^５文字目,W5),
62:      modify(1,^は,変換規則)).
```

これにより、”文字数”を調べ、これがゼロ（０）ならば（47行）、すべての文字を変換したことになるで、タスクを”出力”段階へ受け渡す（49行）。このとき作業記憶要素”再認識”は削除する。（48行）。”文字数”がゼロでない場合は、次の”１文字目”から”５文字目”までを再認識し、再び4.3.1 から4.3.7 までのルールと照合を行うため、タスクを”変換規則”とする（51～62行）。

4.4　出力段階

このルールは以下の通りである。

```
403: p(出力,
404:      タスク(^は,出力),
405:      入力段階(^文字集合,X),
406:      入力情報(^１文字目,W1),
407: —>,write('—>'),putlist(X),nl,
408:      remove(2),
409:      remove(3),
410:      modify(1,^は,入力,^位置,1)).
```

これにより、音韻変換段階で変換された”文字集合”（405 行）を、文字列として画面に出力する（407 行）。その後、作業記憶の要素”入力段階”、”入力情報”を削除し（408 ～409 行）、”位置”を初期値の１にし、タスクを”入力”段階へ受け渡す（410 行）。

5．音韻法則の有効性[5]

表１に示した音韻対応法則を共通語（東京方言）の単語に実際に適用して琉球首里方言の単語を生成した

結果について述べる。なお本システムでは、パーソナルコンピュータのメモリの制約のため、現在のところ上記のルールすべてを一括してプログラムされてはいない。しかし、いくつかのルールを組み合わせて、すべてのルールが正しく動作することは確認している。そこで、全ルールを有し本システムと等価な動作をすることが確認されているPrologで構成されたシステム[6]があるので、ここでは、その結果を示す。

入力単語としては以下の2種について検討した。

［ランダム抽出単語］

沖縄語辞典[1]の標準語索引の奇数ページの右上から1語ずつ抜き出した100 個の名詞。対応する首里方言としては、その索引に示されている首里方言のうち音韻的に首里方言に近いものを選んだ。

［基礎語彙］

文献（7）に示されている東京方言と首里方言の基礎語彙対応表の200 語のうち名詞102 語。基礎語彙とは、日常よく使われる基本的な語彙である。

変換結果を以下の5種類に分類して集計した結果を表2に示す。

（1）変換後の単語が音韻的に首里方言と完全に一致する単語。

（2）変換後の単語の一部が首里方言の単語と一致しないものであり、一致しない部分の音韻対応が、文献(1) に例外的規則として記述されているもの。

（3）変換後の単語の中の1音素だけが首里方言の単語と一致しないものであり、一致しない部分の音韻対応については文献(1) に記述されていないもの。

（4）一致しない部分の音韻対応が、明らかに文献(1) の対応法則に反しているもの。

（5）（1）～（3）のいずれにもあてはまらないものであり、著者らには、東京方言と首里方言の語源が異なると思われた単語。

実行結果によれば、音韻対応法則の有効性は、ランダム抽出単語で27%、基礎語彙で50%である。基礎語彙の一致率がランダム抽出単語の一致率より高いことから、「基礎語彙が時間に対する抵抗が強く、それほど変化せずに残る傾向が強い」という言語学的知見が、共通語と首里方言の間でも成立することが確認できる。このことは、すなわち共通語と首里方言とが祖語を同じとすることの傍証となる。

図3に、音韻変換前の単語の一致率と変換後の一致率を示す。同図には、完全一致率の他に、ベンダーの単語一致判定基準[8]による一致率、ＣＶ一致法[8]による一致率も併せて示した。ベンダーの判定基準とは、比較対象の両単語に同一の「子音＋母音＋子音」の並びが存在すれば一致とみなすものであり、ＣＶ一致法とは、両単語に同一の「子音＋母音」（ＣＶ）があれば一致とみなすものである。

図3から、一見無関係と思われる（変換前の完全一致率10数%）琉球方言と東京方言の単語が、音韻対応法則を適用すれば、かなり一致する（基礎語彙の完全一致率50%、ＣＶ一致法では約75%）ことがわかる。ベンダーの判定基準では、ランダム抽出単語の方が基礎語彙より一致率が高く、上記の言語学的知見に反する。これは、日本語がＣＶ型の音節構造を持つからであり、ベンダーの方法は日本語の判定には適さないと考えられる。

他の言語学的研究の結果[9]によれば、首里方言と東京方言の一致率は71.82 %である。ただし、この研究では、一致判定基準は、「語幹の一致」であり、使用した語彙は基礎語彙200 語ではあるが、その中には用言も含まれており、本研究とは異なっている。またその中には、単語中の1音素だけが一致するだけで一

表2．変換結果の分類（個数）

	1)	2)	3)	4)	5)	計
ランダム抽出単語	27	6	8	6	53	100
基礎語彙	51	5	3	4	39	102

1)変換後首里方言と完全に一致した語
2)文献(1)に例外的な音韻変換として挙げられている語
3)不規則変化の単語であるため一音素のみ不一致の語
4)首里方言の方が規則から外れる形で変化した語
5)語源が異なると思われる語

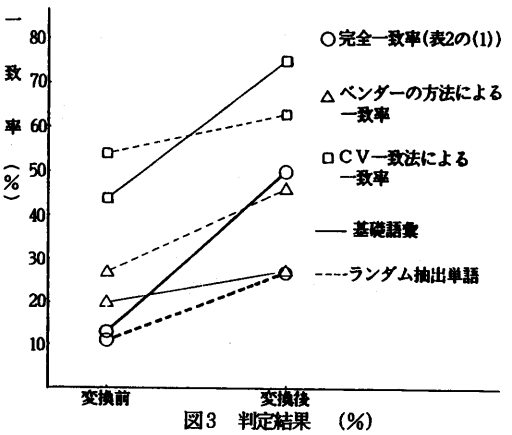

図3　判定結果　(%)

致と見なされているものもある。いずれにせよ、この結果は、本研究における単語変換後のＣＶ一致法による一致率に近い。

以上のように、本システムでは、手作業による方法と同様の結果を得ることができるが、音韻法則を厳格に（確定的なものだけ）適用すれば音韻法則の有効性は50%であるというように、判定基準を明確に示すことができる利点がある。

6．むすび

琉球首里方言と本土方言との間に存在する音韻対応法則をＯＰＳ５のＩＦ－ＴＨＥＮルールとして記述し、共通語の単語から首里方言の単語を生成する単語変換システムを構成した。また、このシステムで用いた音韻対応法則を東京方言の名詞に適用し、生成した単語を実際の首里方言の単語と比較した結果、音韻法則の有効性は基礎語彙で50%であることがわかった。

本システムでは、ＯＰＳ５のＩＦ－ＴＨＥＮルールの構文を利用することにより、音韻法則（規則）を明確に規定することができ、これにより音韻法則の有効性をより客観的に検討することができる。また、本システムのルールを整理・実行する過程を通して、法則の整理・簡略化を実験的に検討することができる。

今後の課題としては、音韻対応の例外的規則も適用できるようにすること、および規則間の競合関係を説明する機構を付加すること、さらにルールをより自然言語に近い形式で挿入・削除できるようにし、一般の言語学者にも容易に利用できるシステムにすることがあげられる。

文献

（１）国立国語研究所編：“沖縄語辞典”，大蔵省印刷局（昭58－04）．

（２）高良，鉢嶺：“琉球方言の語頭声門破裂音の分析”，信学論（Ａ），J69-A,7,pp.921-922（昭61－07）．

（３）鈴木宣夫：“ＯＰＳ５文法入門”，Computer Today （サイエンス社），No.13，pp.11-19（昭61－05）．

（４）小林・小沢：“OPS5 on Prolog”，Computer Today （サイエンス社），No.13，pp.54-60（昭61－05）．

（５）高良，志喜屋，新垣：“音韻法則を用いる共通語から琉球方言への単語変換”，昭62電気関係学会九州支連大，ｐ.454（昭62－10）．

（６）志喜屋　章：“Prolog言語を用いる共通語から沖縄方言の単語翻訳”，琉球大工学部卒業論文（昭62－03）．

（７）安本美典：“日本語の成立”，講談社（昭57－01）．

（８）安本，野崎：“言語の数理”，筑摩書房（昭51－07）．

（９）大城　健：“語彙統計学（言語年代学）的方法による琉球方言の研究”，沖縄文化論叢　５言語編（平凡社），pp.491-521（昭47）．