文書処理とヒューマンインタフェース 16－4
（１９８８．１．１８）

２ストローク入力用仮名漢字変換システム

喜多　辰臣　塩見　彰睦　河合　和久　大岩　元

豊橋技術科学大学

ＴＵＴコードは、無想式２ストローク入力方式と呼ばれる日本語入力方式の１つであり、漢字を直接入力できるので、高速な入力が可能である。しかしＴＵＴコードではコードを覚えていない等の理由より、使用する漢字を直接入力できない場合には、仮名漢字変換を併用することにより、その漢字を入力する。ところが、仮名漢字変換と併用すると、１つの問題が発生する。熟語を入力する際に、その熟語を構成する漢字を１文字でも知らないと、仮名漢字変換では、その熟語の読みをすべて仮名で入力して、変換しなければならず、覚えている漢字のコードが無駄になってしまう。そこで、本稿では、この問題点を解決するための漢字混じり仮名漢字変換の方法を３つ示し、それらを比較した。また、その結果より決定した漢字混じり仮名漢字変換の方法を用いて、日本語フロントエンド・プロセッサとして作成したＴＵＴコードのための仮名漢字変換システムについて報告する。

A Kana-to-Kanji Conversion For Mixed Kana and Kanji Input

T. Kita, A. Shiomi, K. Kawai and H. Ohiwa

Department of Computer and Information Sciences, Toyohashi University of Techology
1-1, Hibarigaoka, Tenpaku-cyo, Toyohashi, Aichi, 440

Although Kanji may be input directly by using a two-stroke code method such as TUT-code, Kana-to-Kanji conversion is necessary when a code is not assigned for a Kanji to be input or cannot be recalled by the input operator. A new conversion system especially suitable for direct input method of Kanji has been developed for the case in which input text is composed not only of Kana but also of mixed Kanji and Kana. Three methods for implementing the system are shown and discussed. A front-end-processor for Japanese text input using this conversion system is also described.

１．はじめに

我々の研究室では、従来から、日本語高速入力方式であるＴＵＴコード[1]を開発し、その応用に付いて研究している。この方式は、当初は、ワード・プロセッサ用の専用機の上で稼働していたが、現在は、パーソナル・コンピュータ（以下、パソコンとする）上の日本語フロントエンドプロセッサ（以下、ＦＥＰとする）に組み込まれて稼働している。

しかし、このＦＥＰは、もともと仮名漢字変換専用のものであり、ＴＵＴコードは、そのＦＥＰに間借りするように、組み込まれているために、様々な不都合が生じている。例えば、ＴＵＴコードの平仮名と片仮名のモード切り替えが、打ちにくいキーに割り付けられている。また、ＴＵＴコードで仮名漢字変換を利用する場合で、仮名を入力しようとして、ミスタイプで、漢字を入力すると、漢字が入力されることが、このＦＥＰでは、漢字＋無変換キーが打たれたと解釈されるために、もう一度仮名を入力し直さなければならない。さらに、ＴＵＴコードと仮名漢字変換を併用すると、ＴＵＴコードが漢字を直接入力できるため、従来の仮名漢字変換を拡張して、漢字混じりの仮名入力に対して漢字変換を行なう必要性も生じた。本稿では、この新しい仮名漢字変換の方法を検討し、ＭＳ－ＤＯＳの使いやすいＦＥＰとして実現した結果について述べる。

２．ＴＵＴコード

ＴＵＴコードは、無想式２ストローク入力方式[2]と呼ばれる日本語入力方式の１つであり、２～３ストロークで漢字が直接入力できるために、高速な日本語入力が可能である。本方式の特徴を次に説明する。

２－１　仮名

ＴＵＴコードでは、仮名をローマ字のように、行と段の組合せで表わし、各行、各段に対応するキーを、行→段の順に打つことによって、仮名を入力する。行段に対応するキーは、図１のように定めている。まず、行を表わすキーを左手で、段を表わすキーを右手で打つように定め、さらに、仮名の使用頻度を基に、よく使うキーが打ちやすい位置（例えば、人差指の中段）に、あるいは使われないキーが打ちにくい位置（例えば、小指の上段）にくるように決めたものである。これによって、仮名文字列の入力は、左手右手左手右手という繰り返しになり、リズミカルな高速入力ができる。例えば、か行あ段の「か」という仮名を入力するには、まず、左手中指上段のか行のキーを打ち、次に、右手中指中段のあ段のキーを打てばよい。以後の説明では、このようなキー入力を、

か＝か＋あ

のように記述する。この式の左辺は、入力した仮名（文字）を表わし、右辺は、まずか行のキーを打ち、あ段のキーを打つことを意味している。したがって、

ひ＝は＋あ

は、「ひ」を入力するには、まず、は行のキーを打ち、次に、い段のキーを打てばよい、ということになる。あ行の仮名は、

あ＝(あ)＋あ

い＝(あ)＋い

のように、あ行を表わす(あ)のキーをまず、打ってから、各段のキーを打つことにする。

以上のように、仮名の入力は、キーボードの３０個あるキーのうち、特に打ちやすい１６個のキーを用い、さらに、キーの割り当てを、行段構造に基づいて覚えやすくしてあるために、１時間に練習で、容易に習得できる。[3]

２－２　漢字

漢字は、２～３ストロークで直接入力可能であり、２ストロークで、７２５字、３ストロークで１８００字を入力できる。また、次の式で、右の文字列を、左の漢字のコードと呼ぶ。

効＝Ｖ＋Ｚ

亜＝Ｔ＋Ｔ＋Ｋ

２ストロークの漢字のコードは、その漢字の頻度と、前後の文字との接続頻度を考慮して、高速に打てるように割り当ててある。漢字のコードは、習得には時間がかかるが（２ストロークの漢字で２０～１００時間）、習得すれば、高速入力が可能となる。

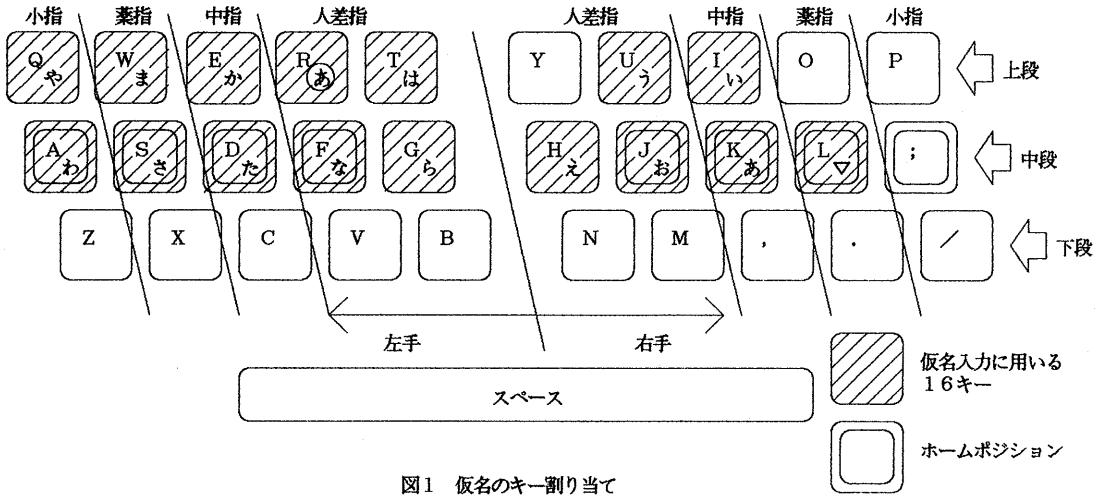


図１　仮名のキー割り当て

３．ＴＵＴコードと仮名漢字変換

ＴＵＴコードは、漢字を直接入力できる方式であるが、次のような場合は、漢字が入力できないために、いわゆる仮名漢字変換と併用しなければならない。

A)入力したい漢字のコードを覚えていない場合
B)入力したい漢字にコードが割り当てられていない場合

A)には、習得した漢字のコードを忘れた場合や、ＴＵＴコードを覚え初めて、日が浅く漢字のコードを覚えていない場合が含まれる。B)の場合は、現在、パソコン等で表示できる漢字が、６０００字以上であり、ＴＵＴコードの２５２５字では、そのすべてを入力することが不可能であるために生じる。しかし、この様に仮名漢字変換とＴＵＴコードを併用しても、仮名漢字変換のみを利用する場合に比べて、高速な入力が期待できる。

ところが、ＴＵＴコードと仮名漢字変換を併用すると、ＴＵＴコードが漢字を直接入力できるために、次のような問題が生じる。いま、ある漢字２文字以上からなるある熟語を入力しようとする場合を考える。その際に、その熟語のある１つの漢字のコードを知らず、他の漢字のコードは知っているとする。この場合、仮名漢字変換を利用することを考えると、通常の仮名漢字変換では、すべて仮名で入力しなければならず、せっかく覚えている漢字のコードを有効に利用できない。例えば、「効率」を入力したいとき、「効」の漢字のコードを覚えていて、「率」の漢字のコードを覚えていないとすると、通常の仮名漢字変換では、「こうりつ」と入力して、変換しなければならず、覚えている「効」のコードが、有効に利用できない。このことは、利用者の漢字を覚える意欲を減退する結果となる。

そこで、この問題を解決するために考えた方法が、漢字混じりの文字列を熟語に変換する漢字混じり仮名漢字変換である。この変換方式は、先ほどの例の場合、「効りつ」という漢字混じりの文字列を「効率」という熟語に変換できる。この変換方式を用いることにより、覚えている漢字のコードが有効に利用できる。

４．漢字混じり仮名漢字変換

４－１　漢字仮名変換[4]

漢字混じり仮名漢字変換を実現する方法として、我々は当初、入力文字列中の漢字を、漢字仮名変換を用いて、読みに変換し、入力文字列を仮名文字列として、その文字列に対して、仮名漢字変換を行なう方法を考えた。図２は、「効りつ」という入力に対する変換の流れを示したものである。この方法の場合は、まず、「効りつ」という漢字混じり文字列が入力されたらその文字列より漢字「効」を切り出して、「効」の読みを漢字かな変換により求める。その結果「き、こう」がえられ、合成された読みとして、「きりつ」「こうりつ」が合成される。これらに対して、仮名漢字変換を行ない。「規律、起立、効率、公立、高率」の候補がえられる。それらの候補と、「効りつ」を比較して、最終的な候補である「効率」をえる

ところが、この方法には、以下の２つの原因により、変換効率が悪くなるという欠点がある。

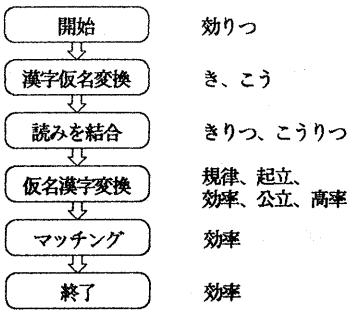


図２　漢字仮名変換を利用する方法

a.入力文字列の読みとは、違う読みも合成する。
b.入力文字列中の漢字が増えると、合成される読みの数が増える。

a.の問題は、漢字の読みが１つであることは、ほとんどなく、音読みや、訓読みを合わせて、複数個の読みが存在するため生じる。このために、入力文字列の読みとして複数の仮名文字列が生成される。そして、それらのすべてを読みとして仮名漢字変換を行なわなければならず、無駄が生じるわけである。

b.の問題は、次の例を考えれば明らかである。例えば入力文字列中に漢字が２字含まれると仮定すると、その漢字それぞれに読みが２つずつあると、入力文字列の読みとして、４つの仮名文字列が生成され、３つ漢字が含まれると仮定すると、仮名文字列は８個生成される。このように漢字の数が増える毎に、合成される仮名文字列の数が、飛躍的に増える。

４－２　新しい方法[5][6]

漢字仮名変換を利用する方法の問題点は、漢字を仮名に変換する際に、漢字の持っている情報が、失われるために発生する。そこで、漢字を変換せずに、そのまま利用して、漢字混じり仮名漢字変換を行なう方法を考えた。考えた方法は、以下に示す２つである。

i)漢和辞典を利用する方法
ii)辞書を拡張する方法

次にそれらを説明する。

４－２－１　漢和辞典を利用する方法

まず、漢和辞典を利用する方法では、あらかじめ漢字に対して、漢和辞典のように、その漢字の用いられている熟語を集めたテーブルを用意する。そして、入力文字列中の漢字で、そのテーブルを検索して、その漢字が使われている熟語中から、入力文字列と比較して、候補を求める方法である。「効率」を例に説明する。まず、入力文字列として、「効りつ」が入力されたとすると、テーブル中で、「効」を用いている熟語（効果、効率、効用、効能等）が納められているレコードを検索して、そのレコード中で、「効りつ」の仮名部分と一致する候補「効率」を求めるわ

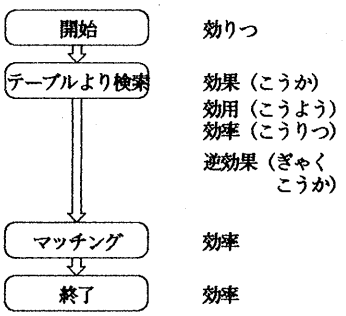


図3　漢和辞典を持つ方法

けである（図3参照）。

4－2－2　辞書を拡張する方法

次に、ii)の方法について説明する。この方法は、通常の仮名漢字変換用の辞書では、見出し語が、仮名より構成されているのに対して、漢字混じり文字列も見出し語とする方法である。つまり、「効率」という熟語を辞書より検索するためには、通常の辞書では、「こうりつ」という見出し語より求めるが、この方法では、「こうりつ」、「効りつ」あるいは、「こう率」で「効率」が求められる。逆に言うと、「効りつ」で検索するには、そのままの文字列で、辞書に検索に行き、「効率」を求めることができるということである。

4－2－3　比較

i)とii)のいずれの方法を、用いるかを検討するために、2つの方法での辞書及びテーブルの大きさを概算する。まず、図4に辞書とテーブルの構造を示す。図4において、

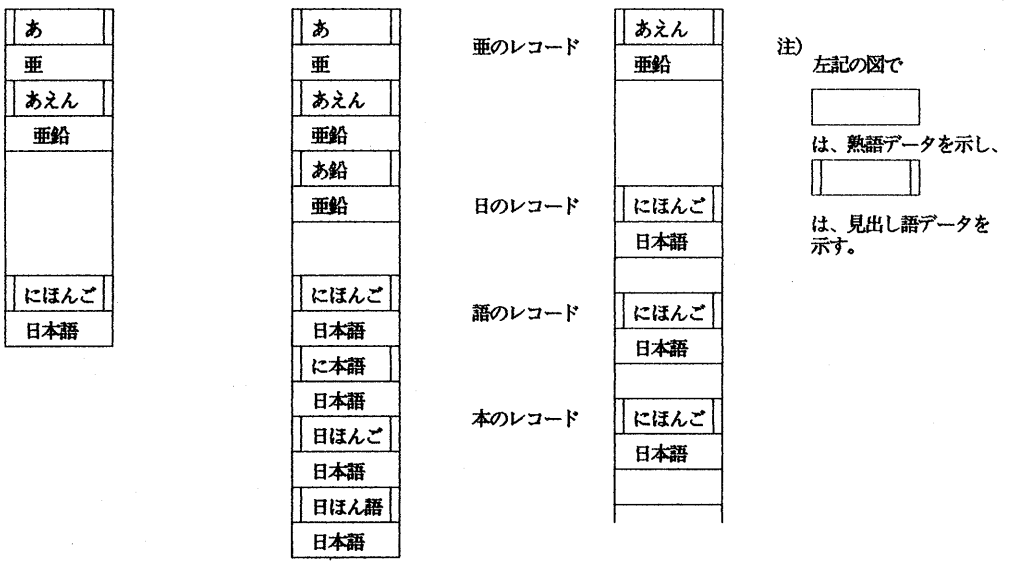
表1　辞書の基礎データ

熟語数	47233語
見出し語の平均文字数	5文字
熟語の平均文字数	2．55文字
熟語中に含まれる漢字数別の熟語数	
漢字数　[文字]	熟語数　[語]
0	3431
1	6825
2	32798
3	4059
4	119
8	1
熟語中に含まれる平均漢字数	1．8文字

Aは通常の仮名漢字変換に必要な辞書、Bはii)の方法のために拡張した辞書、Cはi)の方法のためのテーブルである。ii)の方法では、Bのみで漢字混じり仮名漢字変換も、通常の仮名漢字も、行なえるが、i)の方法では、CのほかにAも通常の仮名漢字変換のために必要である。

表1は、計算に用いた基礎データである。表1のデータより、ii)の方法での辞書の大きさを算出する。辞書の見出し語の数は、熟語中にn文字漢字が含まれると、その熟語に対して、2^n-1個の見出し語が存在する。例えば、漢字が3文字の「日本語」という熟語の場合は、見出し語は、「にほんご」、「日ほんご」、「に本ご」、「にほん語」、「日本ご」、「に本語」、「日ほん語」の7つになる。したがって、拡張した辞書（B）におかれる見出し語の総数は、

$$\begin{aligned} &6825*(2^1-1)\\ +&32798*(2^2-1)\\ +&4059*(2^3-1)\\ +&119*(2^4-1) \end{aligned}$$



A．通常の辞書の構造　　B．拡張した辞書の構造　　C．漢字混じり仮名漢字変換用テーブルの構造

図4　辞書及びテーブルの構造

＋　　１＊（2^8-1）
＝１３９１０３語
となる。漢字が含まれるために、見出し語をシフトＪＩＳコードで表現するとして、見出し語だけで、
１３９１０３＊５＊２
＝約１４００Ｋバイト
必要となり、さらに熟語を収納するために要する大きさが
１３９１０３＊２．２５＊２
＝約６３０Ｋバイト
となることから、合計約２Ｍバイトの容量が必要となる。これは、１枚のフロピーディスクに納まらない大きさである。

次に、i)の方法で、辞書とテーブルの大きさを計算してみる。まず、辞書の大きさは、見出し語が仮名のみのため、仮名１文字をバイトで表現して、
４７２３３＊１＊５＝約２４０Ｋバイト
となり、熟語を収納するために、
４７２３３＊２＊２．２５＝約２２０Ｋバイト
必要であるから、合計、
２４０＋２２０＝４６０Ｋバイト
となる。次に、テーブルの大きさであるが、１つの熟語に平均して漢字が１．８文字含まれるために、１つの熟語は、テーブル中に１．８回存在する。つまり、漢字３文字から成る「日本語」という熟語の場合、「日」、「本」、「語」のテーブルそれぞれに見出し語である「にほんご」と熟語である「日本語」が収納される。即ち、３回表われるわけである。したがって、テーブルの大きさは、辞書の約１．８倍となる。
４６０＊１．８＝約８３０Ｋバイト
この結果i)の方法では、辞書とテーブル合わせて約１．３Ｍバイトとなった。

以上の結果より、大きさから判断すると、ii)の方法では、辞書は、２Ｍバイトと現在のパソコンでは、ハードディスク等大容量の特別な装置を必要とするのに対して、i)の方法では、辞書とテーブルを別々のフロッピーディスクに収納し、利用すれば、例えば、５インチ２ＨＤのフロッピーディスクでも実現可能であり、i)の方法の方が、実用的であると考える。

次に、検索アルゴリズムについて考えると、ii)の方法では、通常の仮名漢字変換と漢字混じり仮名漢字変換が同一のプログラムで行なえるのに対して、i)の方法では、通常の仮名漢字変換と漢字混じり仮名漢字変換それぞれにプログラムが必要となる。

これらの得失に対して、今回作成するシステムが、パソコン上であることを考えて、辞書及びテーブルが小さくなるi)の方法を採用することにした。

５．仮名漢字変換システムの作成

５－１　フロントエンド・プロセッサ（ＦＥＰ）

今回作成したＦＥＰとＭＳ－ＤＯＳシステムの関係を図５に示す。キーボードから入力されたデータは、ＢＩＯＳを通して、ＦＥＰに渡される。そして、そのデータをＦＥＰ内で、様々なデータに加工する。例えば、キーボードで打たれたＴＵＴコードを漢字あるいは仮名にしたり、仮名漢字変換したりする。その加工されたデータをＭＳ－ＤＯＳシステムの渡す。また、ＭＳ－ＤＯＳシステムからの画面表示データは、ＦＥＰを通り、ＢＩＯＳに渡されＣＲＴに表示される。

これまでは、ＴＵＴコードを、利用する多くの場合は、利用するアプリケーションに組み込まれて利用されてきた。この方式では、アプリケーションを作成する毎に、組み込まなければならず、市販のアプリケーションでは、利用できないという欠点があった。そこで、上述のようにＦＥＰとして作成した。この結果多くのアプリケーションにおいて、プログラムを変更することなしに、ＴＵＴコードを利用できるようになった。

５－２　プログラムの作成方法

今回本システムを作成するために、アセンブラ言語とＣ言語の二つを使用し、両者で書いたプログラムをリンクして、動作させることにした。この様にした理由は、以下の通りである。

1. 本システムでは、ＦＥＰをデバイス・ドライバとして、実現しており、そのために必要なデバイス・ヘッダが、Ｃ言語では、記述が不可能である。
2. デバイス・ドライバとして、必要な様々なルーチンにおいて、セグメントの管理が必要であるが、Ｃ言語では、管理が不可能である。
3. 本システムのメインルーチンである変換等のルーチンは、アセンブラで記述すると、作成、デバッグ、保守が困難である。

これらの理由のために、本システムは、アセンブラ言語で書かれたプログラム中からＣ言語で書かれたプログラムを呼び出すこととした。以上のことを言い替えると、デバイス・ドライバをアセンブラ言語で記述し、そのうち仮名漢字変換を行なう部分だけをＣ言語で記述した。この結果デバイス・ドライバ作成と、仮名漢字変換プログラムの作成を切り放すことが出来、効率よくプログラミングが行な

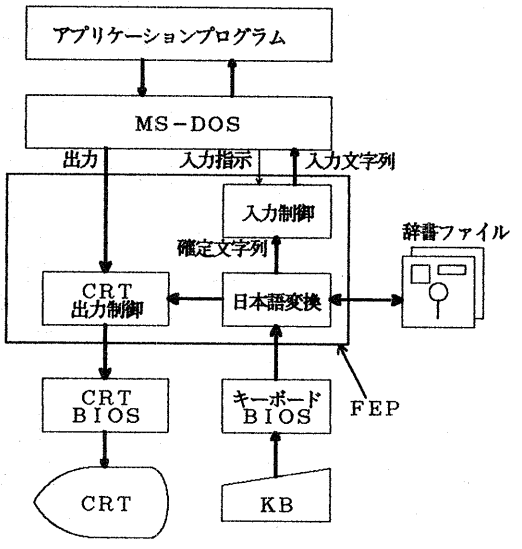


図５　ＦＥＰの構成

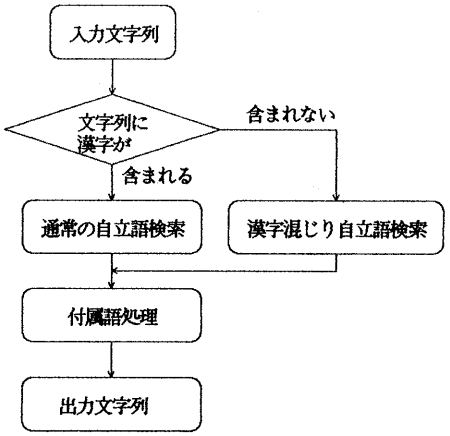


図6　仮名漢字変換の構成

えた。

５－３　仮名漢字変換プログラム

本システムでは、仮名漢字変換方法として、文節変換方式を採用した。本システムにおける文節とは、以下の式で表わせるものとする。

文節＝自立語＋ｎ個の付属語（ｎ≧０）

文節がこの様に構成されているために、文節変換は、自立語検索部と付属語処理部に分けることができる。本システムでは、漢字混じり仮名漢字変換と通常の仮名漢字変換を両立させるために、自立語検索部を、２つに分けた。それらは、入力文字列に漢字が含まれない場合のための通常の自立語検索と、漢字が含まれる場合のための漢字混じり自立語検索である。この自立語検索と付属語処理の流れを図６に示す。仮名漢字変換をこの様に構成した結果、入力された文字列中に、漢字が含まれていなければ、通常の仮名漢字変換が、含まれていれば、漢字混じり仮名漢字変換が自動的に起動し仮名漢字変換を行なう。

５－４　漢字混じり仮名漢字変換の実現方法

４章で漢字混じり仮名漢字変換の手法について説明したが、ここでは、実際の漢字混じり仮名漢字変換用のテーブルの構造とその検索方法について説明する。図７にテーブルの詳細な構造を示す。

テーブルは、可変長のレコードより構成されるテーブル本体と、そのレコードを指すポインタが集まったインデックスより構成されている。１つのレコードは、１つの漢字に対して、その漢字の使われている熟語を集めたものである。インデックスは、漢字のシフトＪＩＳコード順に、その漢字のレコードの位置を並べたものである。レコードは、サブレコードより成るレコード本体と、サブレコードの最初の見出し語と、その位置を示すポインタから成るレコードインデックスを持つ。

レコード本体のデータは、見出し語と熟語とその熟語の情報から構成される。見出し語は、仮名を表わす１バイトのコードと漢字を表わすコードとセパレータによって表現されている。これらは、図中においてそれぞれ、仮名とｘとｉで表わされる。漢字を表わすコードは、「効」のレコードの場合「効率」の見出し語を「ｘｉりつ」と表現して、「効」がｘの部分に当てはまることを示す。つまり、「ｘｉりつ」は、「効ｉりつ」であることを示している。また、

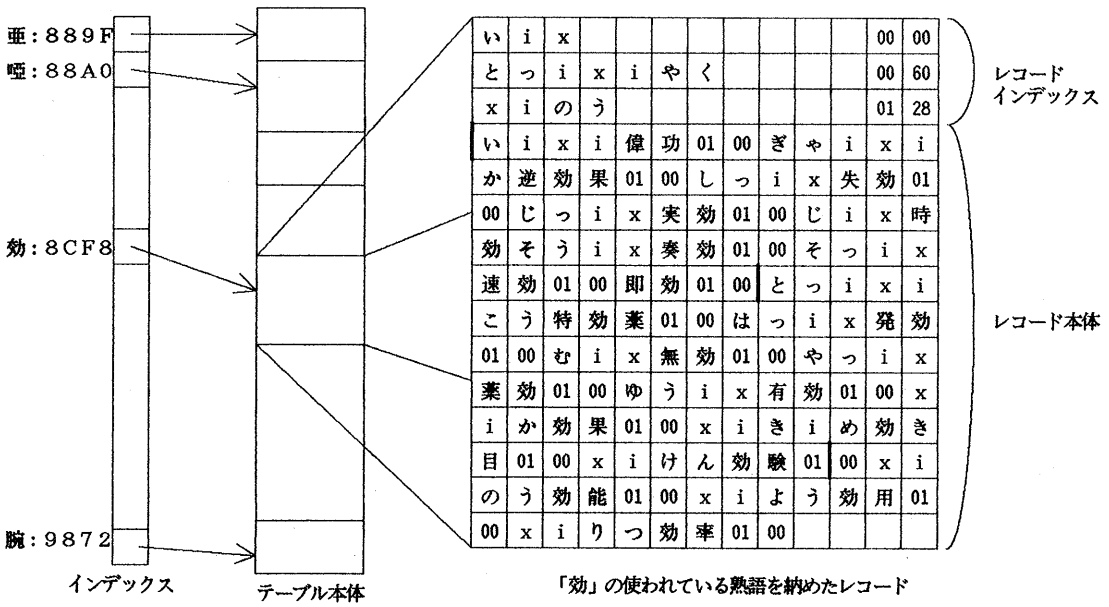


図7　漢字混じり仮名漢字変換用のテーブルの構造

セパレータは、熟語「効き目」の見出し語「ｘｉきｉめ」の場合、熟語を構成する漢字毎の読みに見出し語を分割している。そして、熟語の情報は、２バイトのデータより構成されており、１バイト目は、品詞情報を、２バイト目は、同音異義語選択のための使用時の頻度情報を示す。

検索方法を「効りつ」を例に説明する。入力文字列として、「効りつ」が入力されたとすると、その中から漢字「効」が切り出される。そして、インデックスより、「効」のレコードを求める。そして、レコードインデックスより、「効りつ」が、含まれているサブレコードを求めて、そのサブレコード内を検索して、「効率」を候補として求める。

６．変換用テーブルの作成時の問題点

変換用テーブルは通常の仮名漢字変換用辞書からいかに述べる方法によって、漢字毎にその漢字が使われている熟語を集めたレコードを作成した。このためには、各熟語の漢字毎に、見出し語を分割するセパレータを挿入する必要がある。例えば、「効率」の見出し語「こうりつ」の場合は、「こう」と「りつ」の間にセパレータを挿入する。これによって、「効」、「率」の読みが「こう」、「りつ」であることが分かる。この作業を辞書に収録するデータすべてに対して行なった。この際、以下にあげる点が問題となった。

a)見出し語を分割できない場合
　例）今日、大和
b)一部見出し語が分割できない場合
　例）今日子、大和絵
c)熟語中の漢字が省略されている場合
　例）一宮（「一の宮」の「の」が省略されている）

まず、a)の場合は、見出し語を分割できない熟語が、「効りつ」のように漢字混じり文字列として入力されることはない、つまり、「今ＸＸ」と漢字混じりで入力しようと考えてもＸＸに当たる仮名の部分が入力する利用者にも分からないため、「今日」が漢字混じり文字列として入力されることはなく、必ずすべて仮名か漢字で入力される。このために、この問題に対しては対策を考える必要がないことが分かった。それに対して、b)の場合は、「今日こ」と入力する場合が考えられるので、解決のために、「今日」を１つの漢字と見なして、熟語の方にもセパレータを挿入することにした。つまり、熟語として「今日ｉ子」、その見出し語として「きょうｉこ」と登録する。このように熟語部分にも、セパレータを挿入することにより、「今日」の部分の読みが「きょう」で、「子」の部分の読みが「こ」であることを示すことにした。この結果、テーブルの「今」のテーブルには、見出し語として「ｘｉｙｉこ」が、熟語として「今日ｉ子」が収納される。ここで、「今」のレコードの場合、ｘは「今」を表現しているが、ｙは、「今」以外の漢字であることを表現している。この見出し語により、「今日子」は、最初の１文字が「今」でつぎも漢字で、最後の漢字の読みが「こ」であることが分かる。そして、「今日こ」と入力された場合は、最初の漢字が「今」で、次が漢字で、最後が「こ」なので熟語として「今日子」が選択される。しかし、このままでは、「今月こ」でも「今日子」が選択されるために、入力文字列の漢字部分と熟語を比較することによりその様な問題を解決している。

最後に、c)の場合は、本システムの利用者の中で、「一」が直接入力できる人は、「一みや」あるいは「一のみや」、「宮」が入力できる人は、「いち宮」あるいは「いちの宮」と入力することが考えられる。そのために、「一宮」の見出し語として「いちｉのみや」と「いちのｉみや」を登録することにし、「一のみや」「一みや」「いち宮」「いちの宮」の見出し語で、「一宮」を検索できるようにした。

７．システムの機能

本システムは、ＴＵＴコードを利用しやすくするために、様々な機能を有する。この節では、これらの機能について解説する。

７－１　間接入力と直接入力

本システムの入力モードには、間接入力モードと、直接入力モードの２つがある。間接入力モードは、常時仮名漢字変換用のバッファにデータが入力される。ここで、データとは、ＴＵＴコードで入力した仮名あるいは漢字のことである。そして、その状態で、変換キーを打てば、通常の仮名漢字変換あるいは、漢字混じり仮名漢字変換が起動される。また、そのデータを様々に加工して、ＭＳ－ＤＯＳシステムに渡すことができる。また、データを加工せずにＭＳ－ＤＯＳシステムに渡すためには、無変換キーを打つ必要がある。これに対して、直接入力モードは、データが入力されると直ちにＭＳ－ＤＯＳシステムに渡される。仮名漢字変換等データを加工したいときには、仮名漢字変換を起動した後に、データを入力して、変換を行なわなければならない。間接入力と直接入力の２つのモードを用意した理由は、ＴＵＴコードの初心者は、仮名漢字変換を多く利用するため、間接入力を利用するだろうと考え、漢字を多く覚えているＴＵＴコードの熟練者は、仮名漢字変換をあまり利用しないと考えられ、逆に無変換キーを打つ操作が余分な操作となり、その操作を無駄と感じるであろうから、直接入力を利用すると考えたためである。

７－２　モード

本仮名漢字変換システムは、モードとして、以下にあげる５つのモードを持っている。

A)全角平仮名
B)全角片仮名
C)半角片仮名
D)全角英数
E)半角英数

これらのモードの遷移図を図８に示す。図８に示すように、切り替え方法には、同一の効果を生じる３種類の操作方法を用意した。１つは、２ストロークのコードに、モード切り替えの機能を持たせて、あるコード、例えば、Ａ＋スペースが打たれたらあるモードに切り替わる方法（図８における細線）である。もう１つは、他のシステム等でよくみられるコントロールキーを用いてモード切り替えを行なう方法（図８における太線）である。最後の方法は、仮名のコードの１ストローク目が、小文字であれば、平仮名が、大文字であれば、片仮名が入力される方法（図８における点線）である。

これらの方法を用意した理由を説明する。この２ストロ

ークのコードにモード切り替えの機能を持たせた理由は、ＴＵＴコードの入力のリズムをモード切り替えにも持たせて、リズムを継続するようにすれば、よりよく入力できるのではないかと考えたからである。しかし、半角英字モードあるいは全角英字モードから、他のモードに切り替える際に、２ストロークのコードによるモード切り替えのみでは、英字の次にスペースを入力することができなくなる。なぜなら、その際にモード切り替えが生じてしまうためである。そこで、全角英字モード、半角英字モードの際は、コントロールキーを用いて、モード切り替えを行なうだけにした。これらのモード切り替え用のキーの割り当ては、ファイルに定義して置くことにより、利用者が容易に変更できるようになっている。また、最後の切り替え方法は、平仮名と片仮名の間の切り替えが、他の切り替より頻繁に発生することが予想されるために、その切り替えが容易に行なえるように考えて作成した。

８．おわりに

本稿では、２ストローク入力方式であるＴＵＴコードのための仮名漢字変換システムの作成について報告した。また、そのシステムの持つ機能について解説した。そして、その中でＴＵＴコードと仮名漢字変換を併用する場合の問題点を示し、その解決策とその実現方法について述べた。

本システムが、作成された結果、様々なアプリケーション上で、ＴＵＴコードが利用できるようになり、ＴＵＴコードの利用が促進されると考えられる。

今後は、モード切り替えにおいて、どの切り替え方式がよいか、どのキーを利用する方がよいか等を検討する予定である。さらに、連文節変換機能の付加作業を進めている。また、文語文への対応も検討中である。

参考文献

［１］　大岩他：日本文タッチタイプ入力の一方式，情報処理学会論文誌，Ｖｏｌ１．２４，Ｎｏ．６，

［２］　山田尚勇：専任タイピスト向きタイプ入力法の研究経過，コンピュータソフトウェア，ＶＯｌ．２，Ｎｏ．１，ｐｐ．５４－６４（１９８５）．
ｐｐ．７７２－７７９（１９８３）．

［３］　大岩他：オーディオ・テープを用いた日本語タッチタイプ入力法の訓練、情報処理学会第３２回全国大会、３Ｔ－２（１９８６）．

［４］　宮崎他：２ストローク入力方式における仮名漢字変換，昭６０電気関係学会東海支部連合大会，４７８（１９８５）．

［５］　喜多他：漢字混じり仮名漢字変換，昭６１電気関係学会東海支部連合大会，５０７（１９８６）．

［６］　喜多他：２ストローク入力方式に適した仮名漢字変換，情報処理学会第３５回全国大会，７Ｓ－３（１９８７）．

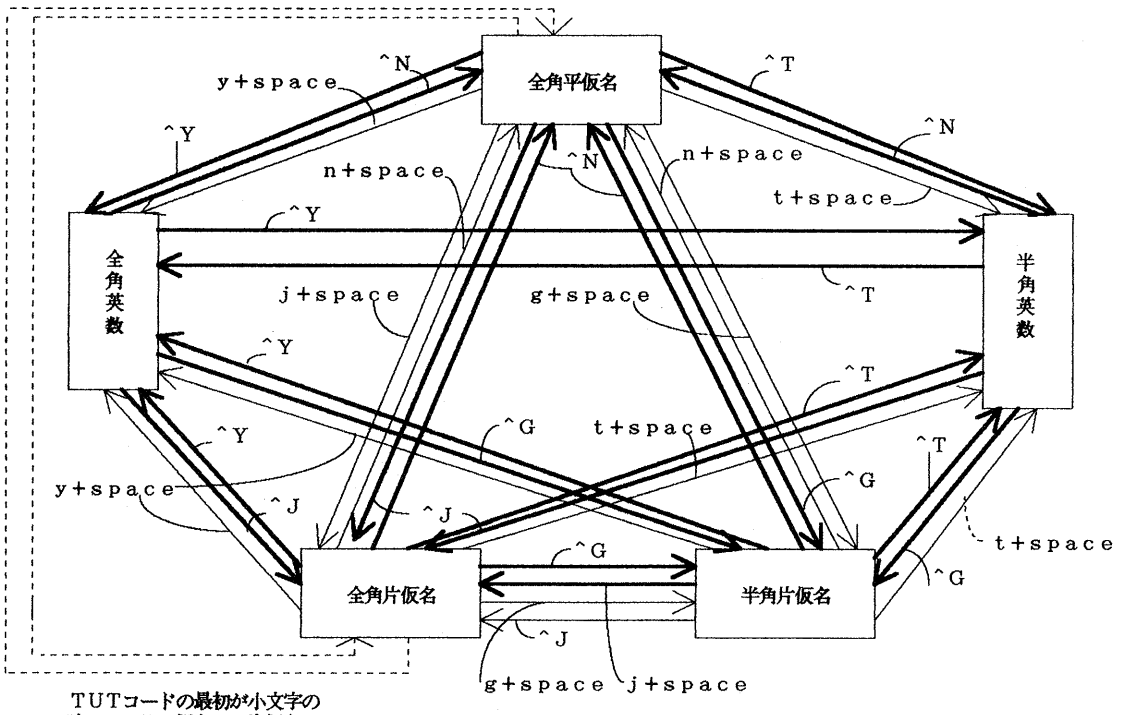


図８　モード切り替え方法