キー順ファイルをベースとした解析／翻訳の実験環境

池田尚志（電総研）中嶋正夫（明治大学）

(1) はじめに

筆者等は、日本語文の構文・意味解析システムや、その応用として機械翻訳システムなどの研究を行っている[1]。自然言語システムでは、辞書データ・ベースをはじめとする大量のデータの管理が重要であり、実験的にシステムを構築していく場合に、データやプログラムの変更・拡張にも容易に対応できなければならない。我々は、辞書データやプログラムをすべてキー順ファイル上に格納し、リスプから直接キー順ファイルにアクセスできるようにしたことで、プログラムやデータの簡便な管理を実現した。本報告では、このキー順ファイルをベースとした実験環境について述べる。

(2) リスプからのキー順ファイルへのアクセス機能

我々はシステムの作成にUTIlisp[2]を用いているが、UTIlisp は、順編成ファイル（および区分順編成ファイル）へのアクセス機能しか備えておらず、大量の辞書データを扱うには不便があった。そこでまず、VSAMのキー順ファイルを扱えるように機能を拡張した[3]。（以下VSAM版という。）

VSAMはディスク装置と主記憶装置の間のデータの転送を行うサービス群であるが[4]、VSAMが扱うファイルのひとつにキー順ファイルがある。キー順ファイルのレコードはキーをもち、各レコードはキーの昇順に配列されている。VSAMは、キー順ファイルに対して、キーを指定することによって直接そのレコードにアクセスしたり、指定されたキーを始点として順次にアクセスするなど種々のアクセス法を提供している。

VSAM版は、キー順ファイルにリスプから直接アクセスできるように、一連のリスプの組込関数を追加したものである。（この機能拡張にはPETL[5]を参考にした。）　追加した関数を（表1）に示す。

アクセス・モードには直接アクセス、スキップ順アクセス、順アクセスの3 種がある。また、代替インデクスを作成して、代替キーによってアクセスすることもできる。直接アクセスおよびスキップ順アクセスの場合には、次のようにキーとエリア番号を指定してア

STREAMDB	VSAM用ストリームの作成	OUTOPENDB	オープン（出力モード）
STREAMDBP	VSAM用ストリームの判定	CLOSEDB	クローズ
INOPENDB	オープン（入力モード）	MODEDB	モードの設定
INOUTOPENDB	オープン（入出力モード）		
直接アクセス、スキップ順アクセス		順アクセス	
PUTDB	S式の出力	PRINTDB	S式の出力
GETDB	S式の入力	READDB	S式の入力
REMDB	キーの削除	DELDB	キーの削除
KEYDB	キーの確認		
GETDBA	S式の入力（代替キーの場合）	その他	
GETDBN	同上（同じ代替キーをもつものが複数あるとき次のものを入力）	POINTDB	順アクセス、スキップ順アクセスの場合の開始キーの位置付け
KEYDBA	キーの確認（代替キーの場合）	POINTDBA	同上（代替キーの場合）
		ENDDB	順アクセス、スキップ順アクセスの終了

(表1)キー順ファイルをアクセスするための関数

クセスする。

(PUTDB stream key area value) ...書き込み

(GETDB stream key area) ...読み込み

key は数値／ストリング／シンボル(漢字も可) であり、areaは 0～255 の整数である。VSAMがアクセスする際の実際のキーは、エリア番号(1バイト) とkey のタイプ(1バイト) およびkey から合成される。キー長はファイル作成時に決まる。

この機能拡張により大きなデータ・ベースをリスプでも容易に扱えるようになった。

(3) 辞書データ・ベース

(3.1) ディスク・ファイル上のデータ構造

現在用いている辞書データの種類を (表2)に示す。これらのデータは、すべて見出しをキーとして

((属性1 属性値1) (属性2 属性値2)...)

の形の属性ー属性値の集合として、辞書ファイルに記述されている。各辞書毎に使用するエリアを定めており、またいくつかの辞書属性毎に異なるエリアに記述しておくこともできる (図1)。

辞書には語彙型の辞書と規則型の辞書がある。語彙型の辞書は、同形異語を区別する必要から設けたもので、同類の語を代表する外部見出しと、内部で区別するための内部見出しをもつ。外部見出しには、

((ICODES IC1 IC2 ...))

の形で、その類に属する語の内部見出しIC1,IC2,... を属性ICODE の値として記述してある。語の諸属性は内部見出しの属性として記述される。

このようにしてすべての辞書属性を１個のキー順ファイル上で管理することができる。エリア０には辞書ファイルを管理するための情報が記述されており、(4) で述べる辞書エディタを通じて辞書ファイルは管理される。

辞書名	見出し	属性
DEF-DICT	辞書名	辞書タイプ、辞書属性、重複禁止エリア
CONST	定数名	定数
SUPSEM	意味特徴名	上位の意味特徴
KRULE	活用型名	活用表
LRULE	接続型名	接続表
CRULE	活用形名	文節カテゴリ規則
PRULE	文節カテゴリ名	係り受け規則(..)、翻訳規則(..)
TRULE	変形規則名	格構造変形規則
ARULE	格役割名	格役割を支える機能語リスト
ANNEX	付属語	語幹、活用型、接続型、意味特徴、整形規則、変形規則、係り受け規則(..)、翻訳規則(..)
TAIGEN	体言	語幹、活用型、意味特徴、係り受け規則(..)、翻訳規則(..)
YOUGEN	用言	同上
SONOTA	その他の自立語	同上
EVERB	英語動詞	過去、過去分詞、... (不規則変化)
ENOUN	英語名詞	複数形、... (不規則変化)
EPRONOUN	英語代名詞	目的格、所有格、 ...(不規則変化)
DVERB	独語動詞	過去、過去分詞、... (不規則変化)
・・・	・・・	・・・

(表2)辞書データベース

エリア	キー	内容
50	イク	((ICODES イク))
51	イク	(($STEM "行" "い"))
52	イク	(($KRULE +IKU) ($SEM <T ...) ($FRULE ((<T)(:T ..)(.TT ..))) ...)
53	イク	(($MRULE ..)($DEP-F ..) ...)
60	イク	(($TL-FRULE*ENG .. go ..) ($TL-MRULE*ENG ..) ...)
61	イク	(($TL-FRULE*DEU .. fahren ..) ($TL-MRULE*DEU ..) ...)
・・・・・		

(図1) キー順ファイル上のデータ

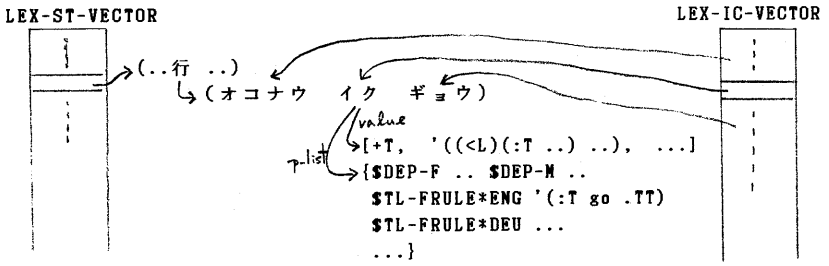
(3.2) 辞書の読み込みと内部構造

辞書情報は内部記憶上では見出しシンボルの値にベクトル値として、あるいは属性リスト上に表現される。ほとんどの見出しに共通する属性はベクトル値として、そうでないものは属性リストで表現しているが、ベクトル値として表現するか属性リストとして表現するかは、辞書読み込み関数の引数で簡単に指定できる。

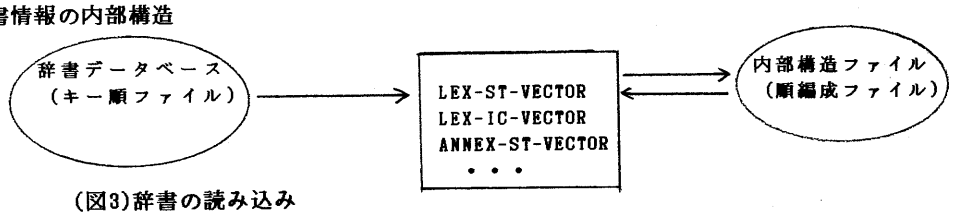
自立語および付属語の語幹と内部見出しシンボルは、通常のオブジェクト・ベクトルとは別のオブジェクト・ベクトル(LEX-ST-VECTOR,LEX-IC-VECTOR,ANNEX-ST-VECTOR,ANNEX-IC-VECTOR) に登録してある。(外部見出しは内部記憶上には不要な情報である。) 語幹のシンボルは、その値として、その語幹をもつ語の内部見出しのリストが設定してある。語幹のオブジェクト・ベクトル(LEX-ST-VECTOR,ANNEX-ST-VECTOR) は語彙解析の段階で語を切り出し同定する際に検索されるオブジェクト・ベクトルとなる。

語の参照はLEX-IC-VECTOR,ANNEX-IC-VECTOR への参照となるから、辞書記述の中で語を参照する必要がある場合には、 ;タロウ、 ,サエ のように自立語の場合には ; を、付属語の場合には , を先頭に付加することによりこのことをマークしている。 ; と , には後続の記号をそれぞれLEX-IC-VECTOR,ANNEX-IC-VECTOR に読み込むようにREAD-MACROが定義してあり、それを有効にするモード(RTDモード)、無効にするモード(RTXモード) を指定できる。

キー順ファイルからの辞書の読み込みは、いくつかのエリアに記述してある辞書属性から内部見出しシンボルの属性を構成しているので、直接アクセス・モードで行なっている。このために、大きな規模の辞書の場合には読み込みに時間がかかる。そこで、RTX モードで生成した内部構造を順編成ファイル (内部構造ファイルとよぶ) にはき出しておいて



(図2)辞書情報の内部構造



(図3)辞書の読み込み

そこから読み込むモードも用意してある。辞書を更新した際には、その更新履歴をとっておいて、この内部構造ファイルも更新することができる。

すべての辞書属性は1 個のキー順ファイルに納めることができるが、内容的には、基本的な規則類と付属語を内容とする核辞書、基本的な自立語を内容とする基本語辞書、特別な分野のための辞書、実験用の一時的な辞書などに分けておくのが実際的である。これらは単に順番に読み込むことによって重ね書きされて追加・更新される。

(4) エディタ

キー順ファイル上のデータを編集するために 2種のエディタを用意している。 1つはキー単位に編集する一般的なエディタ(KEDIT) であり、他の 1つは(3.1) で述べた属性ー属性値の集合のデータ・ファイルをを編集するもの(DEDIT) である。いずれもS式を編集するレベルでは、UTIlisp のライブラリとして開発されている構造エディタAMUSE[6]を用いている。

KEDIT は、キーを指定してAMUSE を呼び出す、キーを削除する、エリア間の移動をするなどのコマンド群をとりそろえたものである。この他、キー順ファイルを確保する、 2つのキー順ファイルをマージする、順編成ファイルに清書出力するなどのコマンド群も用意されている。

プログラムの開発にはKEDIT を用いている。普通、関数名をキーとし、エリア毎にひとまとまりのプログラムを格納している。同名の関数（同名のキー）で版の異なるものを別のエリアに保存しておくことができるなど、プログラムの開発に便利である。

DEDIT は、エリア0に記述してある辞書管理情報を用いて辞書を編集する。エリア0には次のような情報が記述してある。

```
キー：   DNAME
内容：   ((¥DTYPE type area)
          (¥PROPS (prop1 area1 condition1) (prop2 area2 condition2) ...)
          (¥AREAS-R a1 a2 ...)
          (¥COMMENT  ・・・))
```

DNAME は辞書名であり、各辞書の構造は¥DTYPE、¥PROPS、¥AREAS-Rの 3つの属性で記述される。

属性¥DTYPEで、typeは辞書DNAME が語彙型の辞書か規則型の辞書かを区別する。areaは、DNAME のどのエントリももつ属性が入っているエリアである。（エントリが既出のものであるかをみたり、エントリをリスティングする際必要となる。）

属性¥PROPSで、prop はDNAME の名であり、area はその辞書属性を格納するエリア、condition はその辞書属性に対する記述が満足すべき条件である。condition はその属性に対する辞書記述の編集が終り編集モードをぬける時点で、その記述を引数として評価する関数であり、値がNIL になるなら記述の形式に誤りがあるとみなして、その辞書記述を編集するモードに再入する。

属性¥AREAS-Rで、a1, a2, ... は、新しくエントリを登録する場合に、エントリが重複してはならない他の辞書のエリア番号であり、重複していれば警告を出す。

属性¥COMMENTには任意のS式をコメントとして記述できる。

エリア０内のキーの集合が、その辞書ファイル中の辞書名の集合であるが、このエリア０内の記述自身も 1つの規則型の辞書である。これは、DEF-DICTという名前でエリア０に登録してある。DEF-DICTの辞書記述を図4 に示す。

従って、図4 のDEF-DICTがエリア０に最初に記述してあれば、DEDIT を使ってエリア０の内容を編集することにより、任意の辞書および辞書属性を辞書ファイル上に容易に登録することができ、辞書／辞書属性の追加・変更にも簡単に対処することができる。

```
((¥DTYPE RTYPE 0)
 (¥PROPS (¥DTYPE 0 {AND {(CAR X)  (LTYPE RTYPE)}
                        {0 (SECOND X) 256}}))
         (¥PROPS 0 {EVERY X (FUNCTION(LAMBDA(Y)
                    {AND (SYMBOLP (CAR X))
                         {0 (SECOND X) 256}
                         {(THIRD X) は関数のbody}
         (¥AREAS-R 0 {EVERY X (FUNCTION(LAMBDA(Y)
                         {0 (SECOND Y) 256}))})
         (¥COMMENT 0 {T}))
 (¥AREAS-R)
 (¥COMMENT  辞書管理情報)
```

(図4)DEF-DICTの内容

DEDIT では、編集する辞書ファイルの指定、親ファイルの指定、辞書名の指定の後、エントリ（外部見出し）を指定するコマンド、語彙型の辞書の場合は内部見出しを指定するコマンド、辞書属性を指定するコマンドを用いて編集するレコードを定め、AMUSE を用いて編集する。親ファイルを指定しておけば、親ファイル内の記述を参照したり、そこからコピーしてくることができる。編集対象を指定するコマンドの他、編集対象とする辞書属性の範囲の指定、記述内容の一括表示、辞書名やエントリのリスティング、削除、改名、他のエントリの記述の表示やコピー等のコマンドも用意されている。

DEDIT では編集後、その場でそのエントリの記述を内部構造に読み込み、解析／翻訳システムを起動して実験することができる。辞書エディタ上で種々の実験をしながら辞書を作っていくことができるので便利である。

DEDIT で更新／削除したエントリについては、（辞書ファイル名、辞書名、エントリ名）のリストがUPDATE-LIST,DELETE-LIST に保存される。これを用いて内部構造ファイルを更新することができる。

なお、エディタ使用中は、オブジェクト・ベクトルを作業用の一時的なオブジェクト・ベクトルに切り換えておくことによって、メモリ中に無用なデータが蓄積されるのを防ぐことができる。

(5) 入力文の管理

解析／翻訳実験の入力文の管理にもキー順ファイルを使っている。入力文は端末からカナ漢字変換して直接入力することもできるし、ファイルからキーを指定して（あるいはエリアを指定してエリア内の文を順次に）入力することもできる。端末から直接入力した文はキー順ファイル上にメモしておくことができる。また、ファイル上の文に多少の変更を加えたり括弧を挿入するなどの前編集を加えたいことがしばしば生じるが、その場合にはファイル上の文を端末画面上に呼び出し、画面上で編集して入力することができる。編集された文もキー順ファイル上にメモしておくことができる。

このように、キー順ファイルを使って入力文の保存、呼び出し、編集等の管理を容易に行うことができる。

(6) おわりに

リスプからキー順ファイルにアクセスできるように機能を拡張し、また、キー順ファイルに対するエディタを作成した。プログラムの管理、辞書データの管理、入力文の管理などを、すべてキー順ファイルをベースとして行うことにより、簡易で適応性に富む実験環境が実現できた。

現在、このような道具立ての上で、日本語文の解析、翻訳、テキストからの語彙抽出システム、意味表現の研究などを進めている。

参考文献

1)池田、語法規則変換方式による機械翻訳，情報処理学会「自然言語処理技術シンポジウム」，1984
2)Chikayama,T. UTIlisp Manual, METR 81-6, University of Tokyou, 1981
3)池田　リスプによるキー順データセットへのアクセス機能，電総研彙報，48-8，1984
4)富士通　FACOM OS IV/F4 VSAM 機能説明書
5)塚本亨治　Petlシステム説明書，電総研，1982
6)Nakashima,H. Tomura,S. Introduction to AMUSE. A Multi-Use Structure Editor, METR 83-3, University of Tokyou, 1983