

TM-0010

Prologによる日本語文節 DCG の生成

三吉秀夫 向井国昭 平川秀樹
安川秀樹 占川康一

August, 1983

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03) 456-3191~5
Telex ICOT J32964

Institute for New Generation Computer Technology

Prologによる日本語文節DCGの生成

昭和58年8月1日

財団法人 新世代コンピュータ技術開発機構
研究所 第2研究室

三吉秀夫	向井国昭	平川秀樹
友川秀樹	古川康一	

目 次

1. はじめに.....	1
2. 日本語の活用語.....	2
3. プログラムの構成.....	2
3.1 形態素プログラムの概要.....	2
3.2 辞書項目.....	4
3.3 語尾処理述語.....	6
3.4 接続条件.....	8
3.5 基本文法.....	10
4. まとめ.....	13

付録Ⅰ.

- 1.1 日本語動詞の活用
- 1.2 日本語形容詞の活用
- 1.3 日本語形容動詞の活用
- 1.4 日本語助動詞の活用
- 1.5 日本語助動詞の接続条件

付録Ⅱ. 形態素解析プログラム改良版

- 2.1 日本語文節DCG
- 2.2 自立語辞書
- 2.3 付属語辞書
- 2.4 語尾辞書
- 2.5 実行例

1. はじめに

現在 ICOT では本格的な自然言語処理システムのベースとなる日本語／英語の文法を開発中である。我々はすべての自然言語処理システムを 1 階述語論理を基礎とした Prolog [Kowalski 74] を用いてインプリメントし、構文解析のための文法記述も文脈自由文法（以後 CFG と記す）に基く DCG (Definite Clause Grammar) モデルを採用する方針である。その理由として、現在人工知能向言語として注目されている Prolog は自動バックトラック、ユニフィケーションを基本機能として持ち、lisp と同程度あるいはそれ以上の能力を持つ言語であるということ、また従来の言語では複雑な手続きで書かれていた処理がより簡潔に記述できる長所がある点があげられる。また DEC System-10 Prolog に組込まれている DCG というパーサ [Pereira 80] は、CFG 規則が Prolog プログラムの基本的シンタクスである Horn 節との類似性を利用して実装されているもので、文法カテゴリが Prolog の述語として表現され、1 つの文法規則が Prolog の 1 つの Horn 節にコンパイルされていると見ることができる。そのため文法カテゴリに引数を与えたり、文法カテゴリの間に Prolog のプログラムを挿入できるため文脈依存的な情報も扱うことが可能である。また最近の言語学分野では、従来変形を用いて分析を行っていた言語現象を CFG を用いて分析する文法理論も出現しており [郡司 83, Gazdar 81-82], 今後の自然言語処理の研究の流れとして CFG を基礎とする文法を論理型プログラミング言語でインプリメントするのが有力と考えられるからである。

計算機による日本語の構文解析を行うことを考えた場合、活用の種類が多い、分かち書きの習慣がない、英語に比較して語順の自由度が大きい、同音異義語が多い、省略が多い、等日本語固有の特徴があるため [田中 81], 構文解析の手法に工夫を要する。日本語の活用や助動詞の接続条件については国文法の分野でよく調べられているがこれらをそのまま DCG に組込むならば非常に見づらくなるので大規模な文法を作成する場合、計算機内部の制御構造を意識して開発するのは非現実的である。文法の記述は文法書に書かれている形式に近い形で行い、実際の処理プログラムへの変換はトランスレータやオブティマイザで行うべきである。

このような観点から本稿では日本語文法開発の第一歩として、活用語が連なった「行かせられたくなかったようだ」のような“動詞＋助動詞列”（以後述語句と呼ぶ）の文節に注目し、その形態素解析を行う DCG を半自動的に生成し、いくつかの実験を行ったので報告する。このシステムは次の特徴を持つ。

- (1) 規則的な活用をする動詞については辞書項目生成プログラムにより辞書項目を生成する。規則的な活用をする語尾の処理を行う述語は、語尾処理述語生成プログラムにより生成する。助動詞の接続条件を記述するテーブルを接続条件テーブル生成

プログラムにより生成する。

(2) 文法カテゴリをできるだけ減らす。

そのほか、

(3) 音便処理機能がある。

(4) 自動分かち書き機能を持つ。

2. 日本語の活用語

日本語の用言（動詞，形容詞，形容動詞）および助動詞は次に続く語によってさまざまな活用をする〔橋本80，寺瀬76〕。たとえば動詞に付録 1.1のように五段，上一段，下一段，カ変，サ変，の5種類の活用の種類がある。また形容詞，形容動詞はそれぞれ付録 1.2，付録 1.3のように活用する。助動詞はその活用型が動詞型活用，形容詞型活用，形容動詞型活用，特殊型活用，語形変化なし，の5種類がある（付録 1.4）。また助動詞には上につく語の性質が決まっており付録 1.5のような接続条件がある。

3. プログラムの構成

3.1 形態素解析プログラムの概要

付録 1.1から付録 1.5の文法知識に基づいて述語句の形態素解析を行う文法規則を作成する。まず考えられる方法としては，活用語の語幹および語尾の各活用形に対して別の文法カテゴリを与えて辞書項目として持つというやり方がある。たとえばこの方法をとる松沢等の文法〔松沢81〕では，

(a) “見られる”（「見る」の未然形+「られる」の終止形）

(b) “見られない”（「見る」の未然形+「られる」の未然形+「ない」の終止形）

に対する構文解析結果はそれぞれ図 3.1 (a), (b)のようになる。

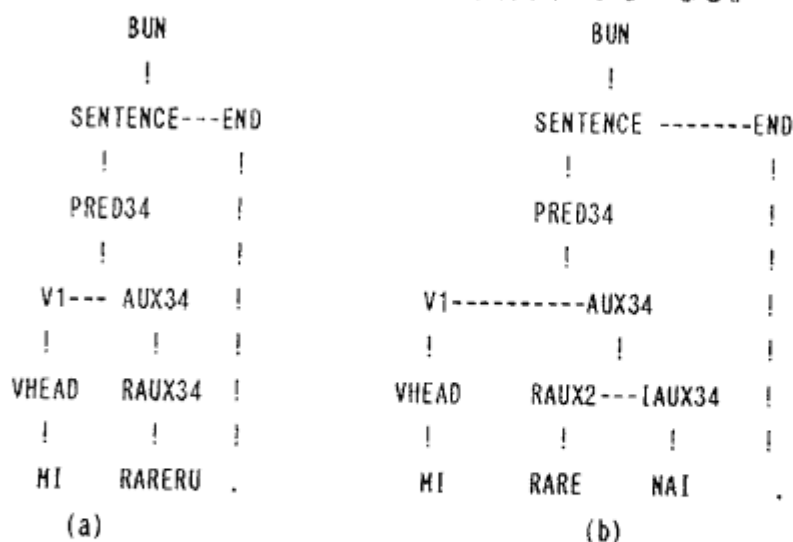


図 3.1

この方法では(a)の“られる”と(b)の“られ”が本来基本形「られる」が変化したものであるにもかかわらず別の文法カテゴリが与えられているため((a)ではRAUX34, (b)ではRAUX2), 別の単語として扱われており, 相互の関連性が表面に表れていない。また扱う活用語が増えるに従って文法カテゴリ数, 辞書項目数が非常に多くなるので, readabilityが悪くなりデバッグも容易でなくなる。従って大規模なシステムを作成するには不向きである。

このような点を改善するために我々は次の方針に従って文法を設計した。

- (i) 語彙カテゴリの数をできるだけ減らす。たとえば我々の文法では“学ぶ”という動詞に対応する辞書項目はDCG形式で

```
word(verb, kak) -->gokan(mana) ,gobi.
```

と表わされるので基本的に語彙カテゴリは語幹gokan とgobiの2つしかない(簡単のため引数をかなり省いている)。

- (ii)規則的な活用をする語については辞書項目生成プログラム(一種のメタルールと考えられる)により辞書項目を作成し, 逐一文法を記述することはできるだけ避ける。

本形態素解析プログラムは次の4つの部分から成る。

- (1) 辞書項目
- (2) 語尾処理述語
- (3) 接続条件テーブル
- (4) 基本文法

(1) – (3) はそれぞれ, 辞書項目生成プログラム, 語尾処理述語生成プログラム, 接続条件テーブル生成プログラムにより生成される。これらの様子を図 3.2に示す。次節より(1) – (4) の項目について説明する。

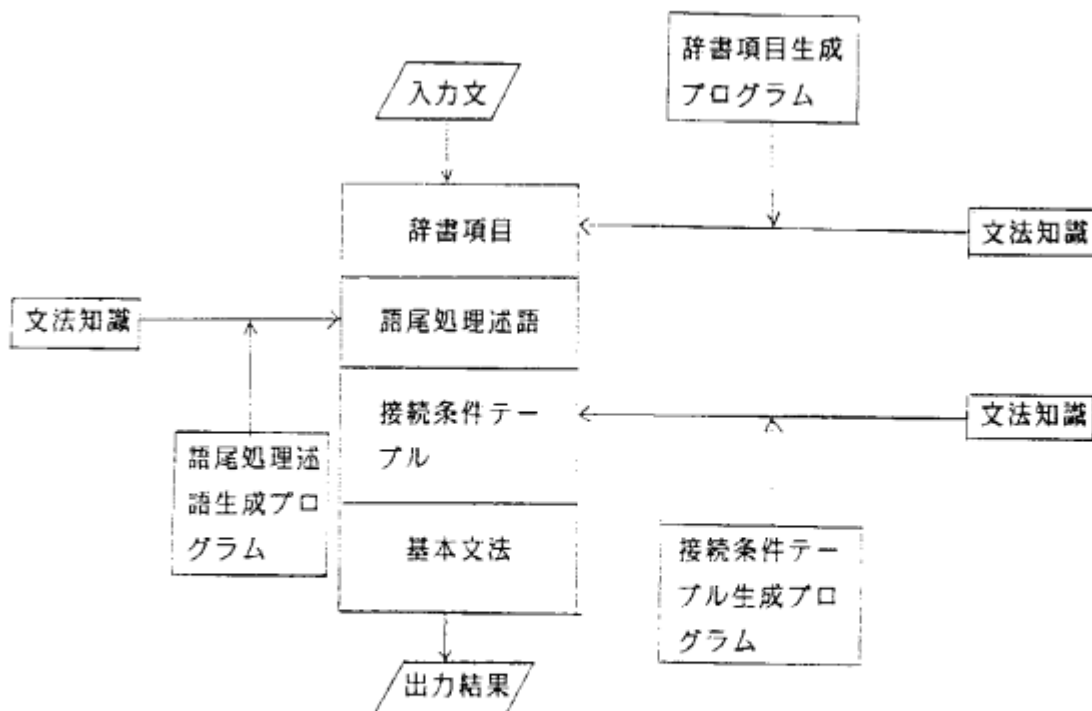


図 3.2 形態素解析プログラムの概要

3.2 辞書項目

3.1でふれたように本プログラムでは活用語である動詞，助動詞の辞書項目は活用しない部分と活用する部分の2つから成るという文法規則として記述される。

たとえば“学ぶ”という動詞は次の形式のDCGで記述される。

```
word(verb, manab, _52, dan_5) --> gokan(mana)
                                gobi(type(verb(dan _5, b)), _52).
```

この規則の左辺の第1引数verbは品詞が動詞であることを示す。第2引数manabは“学ぶ”の代表形（見出し）である。第3引数の変数は解析時に活用形がユニファイされる。第4引数dan _5は五段活用であることを示す。右辺の第1要素gokan(mana)は“学ぶ”の語幹mana（学）を示す。右辺の第2要素gobiが語尾処理を司る述語であり，その第1引数で示される活用の解析を行う。上の例ではバ行五段活用・動詞型の活用語尾の解析をし，第2引数の変数に活用形を返す。述語gobiについては3.3で説明する。動詞に関する辞書項目は次の辞書項目生成プログラムによって生成される。

```

prod_word((word(verb, Word, K, Kname)-->gokan(Stem),
               gobi(type(verb(Kname, Gname)), K))) :-
    (Kname=dan_5,
     (Gname=k, Stem in [o, ka, ma, i];
      Gname=s, Stem in [o, ka, ke, wata];
      Gname=t, Stem in [u, ka, ma, mo];
      Gname=n, Stem in [si];
      Gname=m, Stem in [yo, ku, ka, su];
      Gname=r, Stem in [nobo, kuda, saga, ka, to, ya];
      Gname=w, Stem in [ka, omo, hira, wara];
      Gname=g, Stem in [oyo, ko, huse, ao, nu];
      Gname=b, Stem in [koro, to, mena, hako]);
    Kname=kami_1,
    (Gname=a, Stem in ['', '', o];
     Gname=k, Stem in ['', o, de];
     Gname=t, Stem in [o, ku, mi];
     Gname=n, Stem in ['', ''];
     Gname=h, Stem in ['', ''];
     Gname=m, Stem in [si, '', kaeri];
     Gname=r, Stem in [o, ka, ta];
     Gname=g, Stem in [su];
     Gname=z, Stem in [kan, to, ha];
     Gname=b, Stem in [a, o, no]);
    Kname=simo_1,
    (Gname=a, Stem in ['', tora, kangal];
     Gname=k, Stem in [ma, tasu, sal];
     Gname=s, Stem in [no, ki, mi];
     Gname=t, Stem in [su, a, soda];
     Gname=n, Stem in ['', kasa, tazul];
     Gname=h, Stem in ['', ''];
     Gname=m, Stem in [arata, ho, to];
     Gname=r, Stem in [ku, ka, nagal];
     Gname=g, Stem in [ko, na, ni];
     Gname=z, Stem in [ma, ha];
     Gname=d, Stem in [na, kanal];
     Gname=b, Stem in [sira, ta, nol]);
    Kname=kahen,
    (Gname=k, Stem in ['', ']);
    Kname=sahen,
    (Gname=s, Stem in ['', '];
     Gname=z, Stem in [an])
    ),
    name(Gname, G1), name(Stem, S1),
    append(S1, G1, W),      name(Word, W).

gen_w :- prod_word(W), /* ttynl, display(W),      */
          expand_term(W, Y), fout(Y), fail.
gen_w.

```

このプログラムを「?- gen_w.」で起動することにより図 3.4のような動詞の辞書項目が次々に生成される。

```

word(verb, ok, _43, dan_5, _82, _81):-gokan(o, _82, _130),
                                         gobi(type(verb(dan_5, k)), _43, _130, _81).
word(verb, kak, _43, dan_5, _89, _88):-gokan(ka, _89, _137),
                                         gobi(type(verb(dan_5, k)), _43, _137, _88).
word(verb, mak, _43, dan_5, _92, _91):-gokan(ma, _92, _140),
                                         gobi(type(verb(dan_5, k)), _43, _140, _91).

```


助動詞に関しては例外的な活用をするものが多いので辞書項目を直接記述する。ただし特定の活用型を持つものについては 3.3で述べる述語gobiを語尾処理に用いる。

次に助動詞の辞書項目をいくつか示す。

```
word(auxv, sas+eru, K, verb) --> gokan(sa), gobi(type(verb(simo __1, s)), K).
```

これは使役の助動詞「させる」の辞書項目であり、左辺の第1引数auxvは品詞が助動詞であること、第2引数sas+eruは“させる”の代表形、第3引数の変数は解析時に活用形がユニファイされ、第4引数verbは活用型が動詞型であることを示す。右辺は動詞の辞書項目と同様で、語幹がsaで、サ行下一段・動詞型の活用を示す。

```
word(auxv, na+i, K, adj) --> gokan(na), gobi(type(adj), K).
```

これは否定の助動詞「ない」の辞書項目で、語幹がnaで、形容詞型の活用を示す。

```
word(auxv, soud+a, K, adjv) --> gokan(sou),  
    gobi__1([(renyou, [d, e]), (syuusi, [d, a])], K).
```

これは伝聞の助動詞「そうだ」の辞書項目であり、語幹がsouで、形容動詞型の活用を示している。gobi__1という述語は、例外的な活用語尾処理をするためのもので、この例では、語尾が連用形のdeと終止形のdaしかないことを示す。

3.3 語尾処理述語

3.2で述べたように、語尾処理はgobiまたはgobi__1という述語によって行われ、第2引数に語尾の活用形を返す。

たとえば力行五段・動詞型活用の語尾変化を処理するための述語gobiは次のように定義される。

```
gobi(type(verb(dan__5, k)), mizen) --> [k, a].  
gobi(type(verb(dan__5, k)), renyou) --> [k, i].  
gobi(type(verb(dan__5, k)), syuusi) --> [k, u].  
gobi(type(verb(dan__5, k)), rentai) --> [k, u].  
gobi(type(verb(dan__5, k)), katei) --> [k, e].
```


3.4 接続条件

3.1, 3.2で述べたように、本プログラムでは語尾の各活用形に対して別の文法カテゴリを与えていないので単に、

述語句 --> 動詞+助動詞

という生成規則では「学びます」という正しい述語句だけでなく、「学ぶます」のような正しくないものもアクセプトしてしまう。また、

述語句 --> 動詞+助動詞+助動詞

という生成規則では「学びたいない」のような非文法的なものをアクセプトする。

このような不都合を避けるために、本プログラムでは隣り合う2つの単語の間に接続が可能かどうかチェックするための述語を文法規則中に外部条件として組込んでいる。

付録 1.5によれば各助動詞の前につく語を決定するための統語的な要素には次の5種類がある。

- (i) 品詞（動詞，助動詞，助詞，用言，etc）
- (ii) 活用の種類（五段，上一段，下一段，etc）
- (iii) 活用形（未然形，連用形，終止形，etc）
- (iv) 活用型（動詞型，形容動詞型，etc）
- (v) 特定の単語，あるいは特定の単語以外

入力文を解析中に助動詞が見つかった場合、本プログラムではそれに前接する単語との接続可能性をチェックするために次の形式の'link'という述語を用いている。

```
link(Auxverb, [Hinshi, Katsuyou __shurui, Katsuyou __kei, Katsuyou__type,
              Individual__word]).
```

linkという述語は付録 1.5を基に作られた助動詞の接続テーブルであり、Prologのfactとして与えられている。第1引数の助動詞が第2引数の統語的要素を持つ単語に接続可能であることを示す。linkの呼び出し側は 3.5で述べる基本文法中にPrologの外部条件として組込まれている。

第1引数Auxverb は各助動詞の代表形で、

Auxverb ∈ {re+ru, rer+eru, mas+u, }

である。

第2引数は上記5種類の統語的要素のリストで、

```

Hinshi ∈ {meishi,doushi,keiyoushi,yougen,katsuyougo, ..... }
Katsuyou__shurui ∈ {dan __5,kami__1,simo__1,kahen,sahen }
Katsuyou__kei ∈ {mizen,renyou,syuuusi,rentai,meirei }
Katsuyou__type ∈ {verb,adj,tokushu}
Individual__word ∈ 単語の代表形の集合

```

である。

次に接続テーブルの例をいくつか示す。助動詞「れる」に対する接続条件「五段・サ変の未然形につく」は次のように表される。

```

link(re+ru,[doushi,dan__5,mizen,__309,__356]).
link(re+ru,[doushi,sahen,mizen,__318,__365]).

```

「～以外の---につく」という接続条件の記述に対してはPrologのnot とmemberを用いて表す。たとえば過去・完了の助動詞「た」の接続条件「ぬ、う、よう、まい以外の助動詞の連用形につく」は次のように表現される。

```

link(ta,[jodoushi,__527,renyou,__626,__680) :-
    not(member(__680,[+nu,u,you,mai])).

```

このlinkという接続条件テーブルについてもユーザが直接入力するのではなくて、文法書で述べられている表現にできるだけ近い形をrow dataとして入力し、接続条件テーブルへの変換は接続条件テーブル生成プログラムが行う。

raw dataは次の形式で記述する。

```

aux __con __table (助動詞の代表形, [<接続条件リスト-1>,
    <接続条件リスト-2>,
    . . . . .
    <接続条件リスト-n>] ).

```

第2引数の要素である<接続条件リスト-m>は前記5種類の統語的要素のうちで文法書に属して記述されているものについて, "hinshi", "katsuyou__shurui", "katsuyou __kei", "katsuyou__type", "indiv" というファンクタの引数にした形のもののリストである。

たとえば助動詞「ない」に関する接続条件の記述。

“動詞および助動詞「れる・られる」, 「せる・させる」, 「たがる」の未然形につく。五段動詞「ある」にはつかない。助動詞「たい」「そうだ」「だ」「ようだ」「みたいだ」「らしい」, 形容詞, 形容動詞の連用形につく。”

に対するraw dataをユーザは次のように入力する。

```
aux __conn__table(na+i, [[hinshi(doushi), katsuyou__kei(mizen),
                           indiv(except(aru))],
                      [hinshi(jodoushi), katsuyou__kei(mizen),
                           indiv(re+ru, rar+eru, s+eru, sas +eru, taga+ru)],
                      [hinshi(jodoushi), katsuyou__kei(renyou),
                           indiv(ta+i, soud+a, d+a, you+da, mitaida, rasi+i)],
                      [hinshi(keiyoushi, keiyoudoushi),
                           katsuyou__kei(renyou)]
                      ]
    ).
```

これを接続条件生成プログラムにかけることにより次の形式の「ない」に関する接続条件テーブルが得られる。

```
Link(na+i, [doushi, _2568, mizen, _2583, _2511]):-
    not(member(_2511, [aru])).
link(na+i, [jodoushi, _2782, mizen, _2808, re+ru]).
link(na+i, [jodoushi, _2980, renyou, _3006, ta+i]).
link(na+i, [keiyoushi, _3178, renyou, _3204, _3228]).
link(na+i, [keiyoudoushi, _3364, renyou, _3390, _3414]).
link(na+i, [jodoushi, _3550, renyou, _3576, soud+a]).
link(na+i, [jodoushi, _3748, renyou, _3774, d+a]).
link(na+i, [jodoushi, _3946, renyou, _3972, you+da]).
link(na+i, [jodoushi, _4144, renyou, _4170, mitaida]).
link(na+i, [jodoushi, _4313, renyou, _4339, rasi+i]).
link(na+i, [jodoushi, _4511, mizen, _4537, rar+eru]).
link(na+i, [jodoushi, _4709, mizen, _4735, s+eru]).
link(na+i, [jodoushi, _4907, mizen, _4933, sas+eru]).
link(na+i, [jodoushi, _5105, mizen, _5131, taga+ru]).
```

テーブル中で統語的要素に変数が与えられているものは、文法書に隅に記述されていないもの、つまり何でもよいことを示す。たとえば品詞が動詞であるものの活用型は動詞型であることが決まっているので活用型を指定する必要はない。

3.5 基本文法

このプログラムで用いられている述語句を生成する文法規則を示す。

% GRAMMAR

```
(1) v(X,Y) --> verb(X,Y,Verb_Type).
```

$$(2) \quad v([A:C], [B:D]) \rightarrow \text{verb}([A], [B], \text{Verb_Type}), \text{auxv}(C, D, \text{Verb_Type}).$$

```

(3) .verb([Daihyou_kei],[Katsuyou_kei],[doushi,
                                     Katsuyou_shurui,
                                     Katsuyou_kei,
                                     verb,
                                     Daihyou_kei]) -->
word(verb,Daihyou_kei,Katsuyou_kei,Katsuyou_shurui).

```

```

_ (4) .aux_verb([Daihyou_kei],[Katsuyou_kei],[jodoushi,
_____.                                     '
_____.                                     Katsuyou_kei,
_____.                                     Type,
_____.                                     Daihyou_kei]) -->
word(auxv,Daihyou_kei,Katsuyou_kei,Type).

```

```
(5) auxv([X],[Y],Precedent_Type) --> aux_verb([X],[Y],Aux_Type),
    (link(X,Precedent_Type)).
```

```
(6) auxv([X:X1],[Y:Y1],Precedent_Type) --> aux_vrb([X],[Y],Aux_Type).
      (link(X,Precedent_Type)),auxv(X1,Y1,Aux_Type).
```

この文法の意味は、

- ・述語句は唯1個の動詞あるいは唯1個の動詞と助動詞列から成る（(1),(2)）。
- ・助動詞列は唯1個の助動詞あるいは唯1個の助動詞と助動詞列からなる（(3),(4)）。

なお、(5),(6)は辞書引き時に統語要素のリストを作るための付加的な規則である。

述語句のカテゴリに相当する'v'の第1引数には、解析後に述語句を構成する単語の代表形のリストがアサインされる。(3),(4)の規則の右辺で、{}で囲まれている述語linkが3.4で述べた助動詞の接続条件をチェックする付加手続きである。aux__verbを実行して助動詞が1個みつかった後、左辺の第3引数に直前の単語の統語要素のリストが与えられているので、2つの単語の接続可能性をlinkでチェックしている。つぎにこのプログラムでいくつかの述語句に対して解析を行った結果を示す。

```
Enter the Sentence
|: kaku.
Tango = [kak]
Katsuyou = [syuusi]
Runtime = 6 msec
```

```
Enter the Sentence
|: kakitakunai.
Tango = [kak,+(ta,i),+(na,i)]
Katsuyou = [renyou,renyou,syuusi]
Runtime = 435 msec
```

```
Enter the Sentence
|: kakaseraretakunakattayoude.
Tango = [kak,+(s,eru),+(rar,eru),+(ta,i),+(na,i),ta,+(you,da)]
Katsuyou = [nizen,nizen,renyou,renyou,renyou,rentai,syuusi]
Runtime = 1309 msec
```

4. まとめ

本実験により、中学校の文法書に書かれている程度の文法的制約事項を満たすように形態素処理を行うプログラムが一種のメタルールを用いることにより、比較的ユーザに分り易い形式を用いて記述、生成できることがわかった。大規模な自然言語処理システムを構築することを考える場合、エンドユーザは言語学者のような計算機が専門でない人も含まれるので、エンドユーザが文法的な知識を直接計算機処理に向けた形式で記述するのではなく、文法的な知識はエンドユーザ記述言語で記述し、オブジェクトプログラムへの変換またそれに伴う最適化の手法の導入などは専用のトランスレータ、オブティマイザが行うという形態をとるのが妥当であろう。この観点から、本実験で用いた手法がその手掛りになると考えられる。

なお本実験では助動詞の相互承接に関するチェックをそれに前接している単語の統語的要素のみを用いて行っているが、更に洗練された判定を行うためには、構文論的な考察〔北原81〕も必要であろう。また、辞書項目が増えた場合、辞書引きを効率的に行う工夫が必要である。

本形態素解析プログラムはDEC2060 上で稼働するDEC system-10 Prologですべてインプリメントされている。今後は本プログラムを形態素解析部に組込んだトータルな日本語文法、及びエンドユーザ記述言語を開発し、意味理解に重点をおいた日本語理解システムの開発を行う予定である。

〔参考文献〕

1. 田中穂積, その他: 自然言語処理技術と言語理論, 電子技術総合研究所調査報告 205号, 1981.
2. Kowalski, R.A. : Logic for Problem Solving, Memo No.75, Dept. of Computational Logic, Univ. of Edinburgh, 1974.
3. Pereira, F. and Warren, D. : Definite Clause Grammar for Language Analysis - A survey of the Formalism and a Comparison with Augmented Transition Networks, Artificial Intelligence, 13, pp231-278, May, 1980.
4. 郡司隆男: 日本語におけるコントロール, 情報処理学会, 自然言語処理研究会 35-3, 1983.
5. Gazdar, G. : Unbounded dependencies and coordinate structure, Linguistic Inquiry 12, pp155-184. 1981.
6. Gazdar, G. and Geoffrey, K.P. : Generalized phrase structure grammar: a theoretical synopsis, Indiana Univ. Linguistic Club, Bloomington, Indiana, 1982.

7. 橋本武：やさしい文法，学燈社，1980.
8. 寺瀬光男：中学口語文法，梧桐書院，1976
9. 松沢利彦：拡張LINGOL上での用言の形態素処理に関する研究，情報処理学会自然言語処理研究会26-5，1981.
10. 北原保雄：日本語助動詞の研究，大修館書店，1981.

付録Ⅰ.

活用の種類	基本形	語幹	活 用 形					
			未然形	連用形	終止形	連体形	仮定形	命令形
五段	置く	置	か こ	き い	く	く	け	け
上一段	閉じる	閉	じ	じ	じる	じる	じれ	じよ じろ
下一段	奏でる	奏	で	で	でる	でる	でれ	でよ でろ
カ変	来る	(来)	こ し せ さ	き	くる	くる	くれ	こい
サ変	する	○		し	する	する	すれ	しろ せよ

付録 1.1 日本語動詞の活用

基本形	語幹	活 用 形					
		未然形	連用形	終止形	連体形	仮定形	命令形
高い	高	かる	かっ く	い	い	けれ	○

付録 1.2 日本語形容詞の活用

基本形	語幹	活 用 形					
		未然形	連用形	終止形	連体形	仮定形	命令形
静かだ	静か	だろ	だっ で に	だ	な	なら	○

付録 1.3 日本語形容動詞の活用

助動詞	活用形						活用型
	未然形	連用形	終止形	連体形	假定形	命令形	
せる	せ	せ	せる	せる	せれ	せろ せよ	動詞型
させる	させ	させ	させる	させる	させれ	させろ させよ	動詞型
れる	れ	れ	れる	れる	れれ	れろ れよ	動詞型
られる	られ	られ	られる	られる	られれ	られろ られよ	動詞型
ない	なかる	なかっ なく	ない	ない	なけれ	○	形容詞型
たい	たかる	たかっ たく	たい	たい	たけれ	○	形容詞型
らしい	○	らしかっ らしく	らしい	らしい	○	○	形容詞型
たがる	たがら たがろ	たがり たがっ	たがる	たがる	たがれ	○	動詞型
だ	だろ	だっ で	だ	(な)	なら	○	形容動 詞型
そうだ (様態)	そうだろ	そうだっ そうで そうに	そうだ	そうな	そうなら	○	形容動 詞型
そうだ (伝聞)	○	そうで	そうだ	○	○	○	形容動 詞型
ようだ	ようだろ	ようだっ ようで ように	ようだ	ような	ようなら	○	形容動 詞型
ぬ	○	ず	ぬ (ん)	ぬ (ん)	ね	○	特殊型
た	たろ	○	た	た	たら	○	特殊型
ます	ませ ましよ	まし	ます	ます	ますれ	ませ まし	特殊型
です	でしょ	でし	です	です	○	○	特殊型

付録 1.4 日本語助動詞の活用

助動詞	活用形						活用型
	未然形	連用形	終止形	連体形	假定形	命令形	
う	○	○	う	う	○	○	語形変化なし
よう	○	○	よう	よう	○	○	語形変化なし
まい	○	○	まい	まい	○	○	語形変化なし
てる	て	て	てる	てる	てれ	てろ	動詞型
でる	で	で	でる	でる	でれ	でろ	動詞型
みたいだ	みたい だろ	みたい だっ みたいで みたいに	みたいだ	みたいな	みたい なら	○	形容動詞型
べきだ	べきだろ	べきだっ べきで	べきだ	(べきな)	べきなら	○	形容動詞型
はずだ	はずだろ	はずだっ はずで	はずだ	(はずな)	はずなら	○	形容動詞型
かもしれ ない	○	かもしれ なかっ	かもしれ ない	かもしれ ない	○	○	形容詞型
なければ ならない	○	なければ ならな かっ なければ ならなく	なければ ならない	なければ ならない	○	○	形容詞型

付録 1.4 日本語助動詞の活用 (つづき)

助動詞	接続条件
せる	五段・サ変の未然形につく。
させる	上一・下一・カ変の未然形につく。
れる	五段・サ変の未然形につく。
られる	上一・下一・カ変と助動詞「せる・させる」の未然形につく。
ない	動詞および助動詞「れる・られる」, 「せる・させる」, 「たがる」の未然形につく。五段動詞「ある」にはつかない。 助動詞「たい」「そうだ」「だ」「ようだ」「みたいだ」 「らしい」, 形容詞, 形容動詞の連用形につく。
たい	動詞型活用の連用形につく。
らしい	名詞・副詞, 動詞型活用・形容詞型活用・特殊型助動詞「ぬ」 「た」の終止形, 助詞「の」「から」「まで」「へ」などにつく。
たがる	動詞型活用の連用形につく。
だ (断定)	名詞・副詞・ある種の助詞につく。 活用語には「の」を介してつく 未然形に「う」のついた「だろう」と, 仮定形「なら」は, 動詞型活用・形容詞型活用・特殊型助動詞「ぬ」「た」の終止形 につく。「なら」だけは, 「ます」にもつく。
そうだ (様態)	動詞型活用の連用形につく。
そうだ (伝聞)	活用語の終止形につく。
ようだ	「の」で終る連体詞・助詞「の」・活用語の連体形につく。ただし 助動詞「う・よう」「まい」「だ」「そうだ」「です」「ます」に はつかない。
ぬ	動詞型活用, および「ます」の未然形につく。サ変には未然形のう ち, 「せ」につく。「ます」につくときは「ん」となる。
だ (過去)	ガ・ナ・バ・マ行の五段動詞につく。
た	用言および, 「ぬ」「う」「よう」「まい」以外の助動詞の連用形 につく。
ます	動詞型活用の連用形につく。

付録 1.5 助動詞の接続条件

助動詞	接続条件
です	名詞・形容詞型活用 of 終止形・ある種の助動詞につく。 未然形に「う」のついた「でしょう」は動詞型活用・特殊型活用助動詞「た」「む」「ます」 of 終止形につく。
う	五段活用 of 動詞・形容詞，形容動詞，助動詞「た」「たい」「だ」「ます」「です」 of 未然形につく。
よう	五段活用以外 of 動詞，助動詞「れる・られる」「せる・させる」 of 未然形につく。
まい	五段動詞と助動詞「ます」「たがる」 of 終止形につく。 上一・下一・カ変・サ変と，助動詞「れる・られる」「せる・させる」 of 未然形につく。
てる	動詞 of 連用形につく。
でる	動詞 of 連用形につく。
みたいだ	体言につく。
べきだ	動詞 of 終止形につく。
はずだ	用言 of 連体形につく
かもしれない	動詞・形容詞 of 終止形につく。
なければならない	動詞 of 未然形につく。

付録 1.5 助動詞の接続条件（つづき）

付録Ⅱ．形態素解析プログラム改良版

本文で述べた形態素解析プログラムは辞書引き等の効率化を図っていないため処理速度が遅かったが、いくつかの工夫を加えた結果約4倍スピードが向上し、またプログラム全体が単純な構成になった。工夫の主な点は次のとおりである。

- (1) 自立語、付属語、語尾の3種類の辞書を用意した。
- (2) 付属語辞書に活用接続条件を記述した。
- (3) 活用接続条件の簡単な記述言語を設けて規則活用と不規則活用を統一的に扱った。
- (4) 辞書引きはDEC10 Prologのハッシング機能を用いて効率化を行った。

プログラムのほとんどすべてを占める上記の辞書は従来の辞書から簡単な変換プログラムにより編集した。

改良版では次の基本的考え方を強調した構成になっている。

- (1) (日本語の)文節は自立語といくつかの付属語が連なったものである。

<文節> ::= <自立語> <付属語>*

- (2) 語幹は音便形を含めて不変部分と定義する。
- (3) 単語は次の4つの属性を持つ：
語幹、識別子(見出し)、品詞、活用型(下一段, etc)
- (4) 付属語はさらに活用接続条件を持つ。活用接続条件はその単語の左に隣接できる単語に関する制約式である。
- (5) 各単語は表層上語幹と語尾からなる。語尾はその単語の活用型と活用形のみから決まる。

辞書の構造

現在は自立語、付属語、語尾の辞書を持ち、形態素解析に必要な情報のみを扱っている。各辞書は次のフォーマットをしている。

- (a) 自立語辞書：

zdict(語幹, 見出し, 品詞, 活用型).

(b) 付属語辞書：

hdict(語幹，見出し，品詞，活用型，活用接続条件)。

(c) 語尾辞書：

gdict(語尾の綴り，活用型，活用形)。

これはすぐ上の(5)の仮定の表現でもある。

以下，

2.1 日本語文節DCG

2.2 自立語辞書

2.3 付属語辞書

2.4 語尾辞書

2.5 実行例

の順にリストを載せる。

2.1 日本語文節 DCG

% A DCG for 'bunsetsu' of Japanese

```
:-public member/2.
:-public bunsetsu/4.
:-mode bunsetsu(-,-,+,?).
```

```
bunsetsu([W|R],LW)-->
    {timer},
    ziritu_go(W),
    huzoku_bu(W,R,LW),
    {runtime}.
```

} 文節は自立語と
付属部から成る。

```
huzoku_bu(C,[],C)-->"",[end_cond(C)].
huzoku_bu(PW,[W|T],LW)-->
    huzoku_go(PW,W),
    huzoku_bu(W,T,LW).
```

} 付属部は 0 個以上
の付属語から成る。

```
ziritu_go(W)-->dict_scan(zdict,_,W).
```

```
huzoku_go(PW,W)-->dict_scan(hdict,PW,W).
```

```
dict_scan(Sw,word(Let,Cat,Kt,Kf),word(M,H,Ktype,Kform),X,Y):-
    append(Z,U,X),
    name(Name,Z),
    (Sw==zdict,
     zdict(Name,M,H,Ktype),
     Exp=true;
     Sw==hdict,
     hdict(Name,M,H,Ktype,Exp)),
    eval(Exp,Kt,Kf,Let,Cat),
    gtrunk(Ktype,Kform,U,Y).
```

} 最長一致法で辞書
引きを行う。
付属語の接続チェック
も同時に 行う。

```
gobi_1([(K,X)|_],X)-->string(X).
gobi_1([_|R],K)-->gobi_1(R,K).
```

```
string([])-->"".
string([X|R])-->[X],string(R).
```

```
gtrunk(reg_type(Type),Kform,X,Y):-
    append(A,Z,X),length(A,L),L<5,
    name(B,A),
    gdict(B,Type,Kform).
gtrunk(irreg_type(X),Kform)-->gobi_1(X,Kform).
```

} 語尾処理を行う。

```
end_cond(_).
```

```
% Evaluate a logical expression with the given
% environment.
```

```
eval(true,_,_,_,_):-!.
eval(k(Ktype),Kp,_,_,_):-!,compatible(Ktype,Kp).
eval(f(Kform),_,Kform,_,_):-!.
eval(w(Midasi),_,_,Midasi,_):-!.
eval(h(Hinsi),_,_,Hinsi):-!.
eval(unless(List),_,_,M,_):-!,\+ member(M,List).

eval(not(A),X,Y,Z,U):-!,
```

} 付属語の接続制約
条件を評価する。

```

\+ eval(A,X,T,Z,U).
eval((A;B),X,Y,Z,U):-
    eval(A,X,Y,Z,U),!,eval(B,X,Y,Z,U).
eval((A,B),X,Y,Z,U):-
    eval(A,X,Y,Z,U),eval(B,X,Y,Z,U).
eval((A->B),X,Y,Z,U):-
    eval(A,X,Y,Z,U),!,eval(B,X,Y,Z,U).
eval((A->B),_,_,_,_):-!.

compatible(X,reg_type(verb(Y,_))):-!,X=Y.

member(X,[X|_]).
member(X,[_|Y]):-member(X,Y).

append([X|Y],Z,[X|U]):-append(Y,Z,U).
append([],X,X).

timer:-statistics(runtime,_).
runtime:-statistics(runtime,[_,T]),
    display('runtime='),display(T),
    display(' ms'),nl.

```

2.2 自立語辞書

```
:-public zdict/0.
:-mode zdict(+,-,-,-).
```

```
zdict(a,abiru,verb,reg_type(verb(kani_1,b))).
zdict(a,abu,verb,reg_type(verb(kani_1,b))).
zdict(' ',aeru,verb,reg_type(verb(sino_1,a))).
zdict(' ',airu,verb,reg_type(verb(kani_1,a))).
zdict(aka,akai,adj,reg_type(adj)).
zdict(akaru,akarui,adj,reg_type(adj)).
zdict(an,anzu,verb,reg_type(verb(sanen,z))).
zdict(ao,aogu,verb,reg_type(verb(dan_5,g))).
zdict(ao,aoi,adj,reg_type(adj)).
zdict(arata,arazameru,verb,reg_type(verb(sino_1,m))).
zdict(arigata,arigatai,adj,reg_type(adj)).
zdict(a,aru,verb,reg_type(verb(dan_5,r))).
zdict(atataka,atatakai,adj,reg_type(adj)).
zdict(c,ataru,verb,reg_type(verb(sino_1,c))).
zdict(atsu,atsui,adj,reg_type(adj)).
zdict(' ',au,verb,reg_type(verb(kani_1,a))).
zdict(de,dekiru,verb,reg_type(verb(kani_1,k))).
zdict(de,deku,verb,reg_type(verb(kani_1,k))).
zdict(genki,genki,adv,reg_type(adv)).
zdict(hako,hakobu,verb,reg_type(verb(dan_5,b))).
zdict(haya,hayai,adj,reg_type(adj)).
zdict(ha,hazoru,verb,reg_type(verb(sino_1,c))).
zdict(ha,haziru,verb,reg_type(verb(kani_1,n))).
zdict(ha,hazu,verb,reg_type(verb(kani_1,z))).
zdict(' ',heru,verb,reg_type(verb(sino_1,n))).
zdict(hiro,hirowu,verb,reg_type(verb(dan_5,w))).
zdict(' ',hiru,verb,reg_type(verb(kani_1,h))).
zdict(ho,homeru,verb,reg_type(verb(sino_1,m))).
zdict(' ',hu,verb,reg_type(verb(kani_1,h))).
zdict(huse,husegu,verb,reg_type(verb(dan_5,g))).
zdict(i,iku,verb,irreg_type(['renyou','116'])).
zdict(kaeri,kaeriniru,verb,reg_type(verb(kani_1,m))).
zdict(kaeri,kaerimu,verb,reg_type(verb(kani_1,m))).
zdict(ka,kaku,verb,reg_type(verb(dan_5,k))).
zdict(ka,kamu,verb,reg_type(verb(dan_5,n))).
zdict(kana,kanaeru,verb,reg_type(verb(sino_1,d))).
zdict(kanga,kangaaeru,verb,reg_type(verb(sino_1,a))).
zdict(kan,kanziru,verb,reg_type(verb(kani_1,c))).
zdict(kan,kanzu,verb,reg_type(verb(kani_1,c))).
zdict(ka,kareru,verb,reg_type(verb(sino_1,r))).
zdict(ka,kariru,verb,reg_type(verb(kani_1,r))).
zdict(ka,karu,verb,reg_type(verb(dan_5,r))).
zdict(ka,karu,verb,reg_type(verb(kani_1,r))).
zdict(kasa,kasanaru,verb,reg_type(verb(sino_1,n))).
zdict(ka,kasu,verb,reg_type(verb(dan_5,s))).
zdict(ka,katu,verb,reg_type(verb(dan_5,c))).
zdict(ka,kazu,verb,reg_type(verb(dan_5,w))).
zdict(henkou,henkou,adv,reg_type(adv)).
zdict(ke,kesu,verb,reg_type(verb(dan_5,s))).
zdict(kihiro,kihiro,adj,reg_type(adj)).
zdict(' ',kiru,verb,reg_type(verb(kani_1,k))).
zdict(ki,kiseru,verb,reg_type(verb(sino_1,s))).
zdict(ko,kogeru,verb,reg_type(verb(sino_1,g))).
zdict(ko,kogu,verb,reg_type(verb(dan_5,g))).
```

zdiot(koro, korobu, verb, reg_type(verb(dan_5, b))).
 zdiot(' ', ku, verb, reg_type(verb(kanen, k))).
 zdiot(' ', ku, verb, reg_type(verb(kami_1, k))).
 zdiot(kuda, kudaruru, verb, reg_type(verb(dan_5, r))).
 zdiot(ku, kumu, verb, reg_type(verb(dan_5, m))).
 zdiot(ku, kureru, verb, reg_type(verb(simo_1, r))).
 zdiot(kurusu, kurusii, adj, reg_type(adj)).
 zdiot(ku, kutiru, verb, reg_type(verb(kami_1, t))).
 zdiot(ku, kutu, verb, reg_type(verb(kami_1, t))).
 zdiot(ka, makeru, verb, reg_type(verb(simo_1, k))).
 zdiot(ka, naku, verb, reg_type(verb(dan_5, k))).
 zdiot(nana, nanabu, verb, reg_type(verb(dan_5, b))).
 zdiot(na, natu, verb, reg_type(verb(dan_5, t))).
 zdiot(na, nameru, verb, reg_type(verb(simo_1, s))).
 zdiot(' ', niru, verb, reg_type(verb(kami_1, n))).
 zdiot(mi, miseru, verb, reg_type(verb(simo_1, s))).
 zdiot(mi, mitiru, verb, reg_type(verb(kami_1, t))).
 zdiot(mi, mitu, verb, reg_type(verb(kami_1, t))).
 zdiot(no, motu, verb, reg_type(verb(dan_5, t))).
 zdiot(' ', nu, verb, reg_type(verb(kami_1, n))).
 zdiot(na, naderu, verb, reg_type(verb(simo_1, d))).
 zdiot(naga, nagereru, verb, reg_type(verb(simo_1, r))).
 zdiot(na, nageru, verb, reg_type(verb(simo_1, g))).
 zdiot(' ', neru, verb, reg_type(verb(simo_1, n))).
 zdiot(ni, nigeru, verb, reg_type(verb(simo_1, g))).
 zdiot(nikurazai, nikurazii, adj, reg_type(adj)).
 zdiot(' ', niru, verb, reg_type(verb(kami_1, n))).
 zdiot(no, noboru, verb, reg_type(verb(simo_1, b))).
 zdiot(no, nobiru, verb, reg_type(verb(kami_1, b))).
 zdiot(nobo, noboru, verb, reg_type(verb(dan_5, r))).
 zdiot(no, nobu, verb, reg_type(verb(kami_1, b))).
 zdiot(no, noseru, verb, reg_type(verb(simo_1, s))).
 zdiot(' ', nu, verb, reg_type(verb(kami_1, n))).
 zdiot(o, oairu, verb, reg_type(verb(kami_1, a))).
 zdiot(o, oau, verb, reg_type(verb(kami_1, a))).
 zdiot(o, obiru, verb, reg_type(verb(kami_1, b))).
 zdiot(o, obu, verb, reg_type(verb(kami_1, b))).
 zdiot(o, okiru, verb, reg_type(verb(kami_1, k))).
 zdiot(o, oku, verb, reg_type(verb(dan_5, k))).
 zdiot(o, oku, verb, reg_type(verb(kami_1, k))).
 zdiot(omota, omotai, adj, reg_type(adj)).
 zdiot(omo, omowu, verb, reg_type(verb(dan_5, w))).
 zdiot(onazi, onazi, adv, reg_type(adv)).
 zdiot(ooki, ookii, adj, reg_type(adj)).
 zdiot(o, oriru, verb, reg_type(verb(kami_1, r))).
 zdiot(oroaka, oroaka, adv, reg_type(adv)).
 zdiot(o, oru, verb, reg_type(verb(kami_1, r))).
 zdiot(osorozai, osorozai, adj, reg_type(adj)).
 zdiot(o, osu, verb, reg_type(verb(dan_5, s))).
 zdiot(o, otiru, verb, reg_type(verb(kami_1, t))).
 zdiot(o, otu, verb, reg_type(verb(kami_1, t))).
 zdiot(oyo, oyogu, verb, reg_type(verb(dan_5, g))).
 zdiot(saga, sagaru, verb, reg_type(verb(dan_5, r))).
 zdiot(sa, sakeru, verb, reg_type(verb(simo_1, k))).
 zdiot(samu, samui, adj, reg_type(adj)).
 zdiot(si, simiru, verb, reg_type(verb(kami_1, m))).
 zdiot(si, simu, verb, reg_type(verb(kami_1, m))).

zdiot(ai, ainu, verb, reg_type(verb(dan_5, a))).
 zdiot(sira, siraberu, verb, reg_type(verb(simo_1, b))).
 zdiot(siro, siroi, adj, reg_type(adj)).
 zdiot(sizuka, sizuka, adjv, reg_type(adjv)).
 zdiot(soda, sodateru, verb, reg_type(verb(simo_1, t))).
 zdiot('i, su, verb, reg_type(verb(sanen, s))).
 zdiot(su, sugiru, verb, reg_type(verb(kami_1, g))).
 zdiot(su, sugu, verb, reg_type(verb(kami_1, g))).
 zdiot(su, sumu, verb, reg_type(verb(dan_5, n))).
 zdiot(su, suteru, verb, reg_type(verb(simo_7, t))).
 zdiot(suzusi, suzusi, adj, reg_type(adj)).
 zdiot(ta, taberu, verb, reg_type(verb(simo_1, b))).
 zdiot(tadasi, tadasi, adj, reg_type(adj)).
 zdiot(tanosi, tanosi, adj, reg_type(adj)).
 zdiot(ta, tariru, verb, reg_type(verb(kami_1, r))).
 zdiot(ta, taru, verb, reg_type(verb(kami_1, r))).
 zdiot(tazu, tazukaru, verb, reg_type(verb(simo_1, k))).
 zdiot(tazu, tazuneru, verb, reg_type(verb(simo_1, n))).
 zdiot(to, tobu, verb, reg_type(verb(dan_5, b))).
 zdiot(to, tomeru, verb, reg_type(verb(simo_1, n))).
 zdiot(tora, toraeru, verb, reg_type(verb(simo_1, a))).
 zdiot(to, toru, verb, reg_type(verb(dan_5, r))).
 zdiot(to, toziru, verb, reg_type(verb(kami_1, n))).
 zdiot(to, tozu, verb, reg_type(verb(kami_1, n))).
 zdiot(tumeta, tumetai, adj, reg_type(adj)).
 zdiot(tura, turai, adj, reg_type(adj)).
 zdiot(u, utu, verb, reg_type(verb(dan_5, t))).
 zdiot(wara, wararu, verb, reg_type(verb(dan_5, w))).
 zdiot(wata, watasu, verb, reg_type(verb(dan_5, s))).
 zdiot(yawaraka, yawarakai, adj, reg_type(adj)).
 zdiot(yo, yomu, verb, reg_type(verb(dan_5, n))).
 zdiot(yukai, yukai, adjv, reg_type(adjv)).

2.3 付属語辞書

```
(:-public hdiot/5).

(:-mode hdiot(+,-,-,-,-).

hdiot(beki, bekida, auxv,
      irreg_type([(mizen,[100,97,114,111]),
                  (renyou,[100,97,116]),
                  (renyou,[100,101]),
                  (syuusi,[100,97]),
                  (rentai,[110,97]),
                  (katei,[110,97,114,97])]),
      %%%%%%%%%%%
      (h(verb),f(syuusi))).

hdiot(' ', da, auxv,
      irreg_type([(mizen,[100,97,114,111]),
                  (renyou,[100,97,116]),
                  (renyou,[100,101]),
                  (syuusi,[100,97]),
                  (rentai,[110,97]),
                  (katei,[110,97,114,97])]),
      %%%%%%%%%%%
      (h(adv);
       (h(noun);
        (h(pp);
         ((h(auxv),f(syuusi));
          ((h(auxv),w(nu));
           ((h(auxv),w(ta));
            (h(verb),k(dan_5),f(renyou)))))))).

hdiot(de, deru, auxv,
      irreg_type([(mizen,[ ]),
                  (renyou,[ ]),
                  (syuusi,[114,117]),
                  (rentai,[114,117]),
                  (katei,[114,101]),
                  (meirei,[114,111])]),
      %%%%%%%%%%%
      (h(verb),f(renyou))).

hdiot(des, desu, auxv,
      irreg_type([(mizen,[121,111]),
                  (renyou,[105]),
                  (syuusi,[117]),
                  (rentai,[117])]),
      %%%%%%%%%%%
      (f(syuusi);
       (h(noun);
        (h(pp);
         ((h(auxv),f(syuusi));
          ((h(auxv),f(syuusi),w(masu));
           ((h(auxv),f(syuusi),w(nu));
            (h(auxv),f(syuusi),w(ta)))))))).

hdiot(hazu, hazuda, auxv,
      irreg_type([(mizen,[100,97,114,111]),
```

```

(renyou,[100,97,116]),
(renyou,[100,101]),
(syuusi,[100,97]),
(rentai,[110,97]),
(katai,[110,97,114,97])),
~~~~~
((h(adj),f(rentai));
((h(adjv),f(rentai));
(h(verb),f(rentai)))).

hdiac(kamosirena, kamosirenai, auxv,
irreg_type([(renyou,[107,97,116]),
(syuusi,[105]),
(rentai,[105])]),
~~~~~
((h(adj),f(syuusi));
(h(verb),f(syuusi)))).

hdiac(mai, mai, auxv,
irreg_type([(renyou,[ ]),
(syuusi,[1])]),
~~~~~
((h(auxv),f(mizen),w(rareru));
((h(auxv),f(mizen),w(neru));
((h(auxv),f(mizen),w(saseru));
((h(auxv),f(mizen),w(seru));
((h(auxv),f(syuusi),w(masu));
((h(auxv),f(syuusi),w(tagaru));
((h(verb),k(dan_5),f(syuusi));
((h(verb),k(kahen),f(mizen));
((h(verb),k(kami_1),f(mizen));
((h(verb),k(sahen),f(mizen));
(h(verb),k(simo_1),f(mizen)))))))).

hdiac(mas, masu, auxv,
irreg_type([(mizen,[101]),
(mise,[121,111]),
(renyou,[105]),
(syuusi,[117]),
(rentai,[117]),
(katai,[117,114,101]),
(moirei,[105]),
(moirei,[101])]),
~~~~~
f(renyou)).

hdiac(mitai, mitaide, auxv,
irreg_type([(mizen,[100,97,114,111]),
(renyou,[100,97,116]),
(renyou,[100,101]),
(renyou,[110,105]),
(syuusi,[100,97]),
(rentai,[110,97]),
(katai,[110,97])]),
~~~~~
(h(noun);
((h(adj),f(rentai));

```

```

((h(auxv),f(rentai),unless([u,you,nai,desu,masu])));
(h(verb),f(rentai))))).

hdiot(na, nai, auxv, reg_type(adj),
~~~~~
((h(adj),f(renyou)));
((h(adjv),f(renyou)));
((h(auxv),f(mizen),w(rareru)));
((h(auxv),f(mizen),w(raru)));
((h(auxv),f(mizen),w(saseru)));
((h(auxv),f(mizen),w(seru)));
((h(auxv),f(mizen),w(tagaru)));
((h(auxv),f(renyou),w(da)));
((h(auxv),f(renyou),w(mitaida)));
((h(auxv),f(renyou),w(rasii)));
((h(auxv),f(renyou),w(souda)));
((h(auxv),f(renyou),w(tai)));
((h(auxv),f(renyou),w(youda)));
(h(verb),f(mizen),unless([aru])))))))))).

hdiot(nakerebanarana, nakerebanaranai, auxv,
irreg_type([(renyou,[107,97,116]),
(renyou,[107,117]),
(syuusi,[105]),
(rentai,[105])]),
~~~~~
(h(verb),f(mizen))).

hdiot(' ', nu, auxv,
irreg_type([(renyou,[122,117]),
(syuusi,[110]),
(syuusi,[110,117]),
(rentai,[110]),
(rentai,[110,117]),
(katei,[110,101])]),
~~~~~
(f(mizen);
(h(auxv),f(mizen),w(masu)))).

hdiot(ra, rareru, auxv, reg_type(verb(simo_1,r)),
~~~~~
((h(auxv),f(mizen),w(saseru)));
((h(auxv),f(mizen),w(seru)));
((h(verb),k(kanen)));
((h(verb),k(kani_1)));
(h(verb),k(simo_1))))).

hdiot(rasi, rasii, auxv,
irreg_type([(renyou,[107,97,116]),
(renyou,[107,117]),
(syuusi,[105]),
(rentai,[105])]),
~~~~~
(h(adv);
(h(noun);
((h(adj),f(syuusi)));
((h(auxv),f(syuusi)));

```



```

    ((h(auxv),f(syuusi),w(au));
    ((h(auxv),f(syuusi),w(ta));
    ((h(pp),w(he));
    ((h(pp),w(kara));
    ((h(pp),w(made));
    ((h(pp),w(no));
    (h(verb),f(syuusi)))))))).

ndict('', reru, auxv, reg_type(verb(simo_1,r)),
#####
    ((h(verb),k(dao_5),f(mizen));
    (h(verb),k(saken),f(mizen)))).

ndict(sa, saseru, auxv, reg_type(verb(simo_1,s)),
#####
    ((h(verb),k(kaken),f(mizen));
    ((h(verb),k(kami_1),f(mizen));
    (h(verb),k(simo_1),f(mizen)))).

ndict('', seru, auxv, reg_type(verb(simo_1,s)),
#####
    ((h(verb),k(dao_5),f(mizen));
    (h(verb),k(saken),f(mizen)))).

ndict(sou, souda, auxv,
    irreg_type([(mizen,[100,97,114,111]),
    (renyou,[100,97,116]),
    (renyou,[100,101]),
    (renyou,[110,105]),
    (syuusi,[100,97]),
    (rentai,[110,97]),
    (katei,[110,97,114,97])]),
    #####
    (f(renyou);
    ((h(adj),f(syuusi));
    ((h(adjv),f(syuusi));
    ((h(auxv),f(syuusi));
    (h(verb),f(syuusi)))))).

ndict(sou, souda, auxv,
    irreg_type([(renyou,[100,101]),
    (syuusi,[100,97])]),
    #####
    (f(renyou);
    ((h(adj),f(syuusi));
    ((h(adjv),f(syuusi));
    ((h(auxv),f(syuusi));
    (h(verb),f(syuusi)))))).

ndict(ta, ta, auxv,
    irreg_type([(mizen,[114,111]),
    (syuusi,[1]),
    (rentai,[1]),
    (katei,[114,97])]),
    #####
    ((h(adj),f(renyou));
    ((h(adjv),f(renyou));

```

```

      ((h(auxv),f(renyou),unless([mu,u,you,mai])));
      (h(verb),f(renyou)))))).

hdiot(taga, tagaru, auxv, reg_type(verb(r,dan_5)),
      ~~~~~~
      f(renyou)).

hdiot(ta, tai, auxv, reg_type(adj),
      ~~~~~~
      f(renyou)).

hdiot(te, teru, auxv,
      irreg_type([(mizen,[ ]),
      (renyou,[ ]),
      (syuusai,[114,117]),
      (rentai,[114,117]),
      (katai,[114,101]),
      (meirei,[114,114])]),
      ~~~~~~
      (h(verb),f(renyou))).

hdiot(u, u, auxv,
      irreg_type([(renyou,[ ]),
      (syuusai,[ ])]),
      ~~~~~~
      ((h(adj),f(mizen));
      ((h(adjv),f(mizen));
      ((h(auxv),f(mizen),w(da));
      ((h(auxv),f(mizen),w(desu));
      ((h(auxv),f(mizen),w(masu));
      ((h(auxv),f(mizen),w(ta));
      ((h(auxv),f(mizen),w(tai));
      (h(verb),k(dan_5),f(mizen)))))))).

hdiot(you, you, auxv,
      irreg_type([(renyou,[ ]),
      (syuusai,[ ])]),
      ~~~~~~
      ((h(auxv),f(mizen),w(rareru));
      ((h(auxv),f(mizen),w(teru));
      ((h(auxv),f(mizen),w(saseru));
      ((h(auxv),f(mizen),w(teru));
      (h(verb),f(mizen),not(k(dan_5)))))))).

hdiot(you, youda, auxv,
      irreg_type([(mizen,[100,97,114,111]),
      (renyou,[100,97,116]),
      (renyou,[100,101]),
      (renyou,[110,105]),
      (syuusai,[100,97]),
      (rentai,[110,97]),
      (katai,[110,97,114,97])]),
      ~~~~~~
      ((h(adj),f(rentai));
      ((h(adjv),f(rentai));
      ((h(auxv),f(rentai),unless([u,you,mai,da,souda,desu,masu]));
      ((h(pp),w(no));

```

(h(verb),f(rental))))).

2.4 語尾辞書

```

:-public gdict/3.
:- mode gdict(+,?,-).

gdict(ba,verb(dan_5,b),nizen).
gdict(ba,verb(dan_5,b),katei).
gdict(ba,verb(dan_5,b),meirei).
gdict(ba,verb(sino_1,b),nizen).
gdict(ba,verb(sino_1,b),renyou).
gdict(bare,verb(sino_1,b),katei).
gdict(bare,verb(sino_1,b),meirei).
gdict(beru,verb(sino_1,b),rentai).
gdict(beru,verb(sino_1,b),syuusi).
gdict(bayo,verb(sino_1,b),meirei).
gdict(bi,verb(dan_5,b),renyou).
gdict(bi,verb(kani_1,b),nizen).
gdict(bi,verb(kani_1,b),renyou).
gdict(bira,verb(kani_1,b),katei).
gdict(bira,verb(kani_1,b),meirei).
gdict(biru,verb(kani_1,b),rentai).
gdict(biru,verb(kani_1,b),syuusi).
gdict(biyo,verb(kani_1,b),meirei).
gdict(bu,verb(dan_5,b),rentai).
gdict(bu,verb(dan_5,b),syuusi).
gdict(da,adjv,syuusi).
gdict(daro,adjv,nizen).
gdict(dat,adjv,renyou).
gdict(de,adjv,renyou).
gdict(de,verb(sino_1,d),nizen).
gdict(de,verb(sino_1,d),renyou).
gdict(dere,verb(sino_1,d),katei).
gdict(daro,verb(sino_1,d),meirei).
gdict(deru,verb(sino_1,d),rentai).
gdict(deru,verb(sino_1,d),syuusi).
gdict(deyo,verb(sino_1,d),meirei).
gdict(e,verb(sino_1,a),nizen).
gdict(e,verb(sino_1,a),renyou).
gdict(ere,verb(sino_1,a),katei).
gdict(ero,verb(sino_1,a),meirei).
gdict(eru,verb(sino_1,a),rentai).
gdict(eru,verb(sino_1,a),syuusi).
gdict(eyo,verb(sino_1,a),meirei).
gdict(ga,verb(dan_5,g),nizen).
gdict(ga,verb(dan_5,g),katei).
gdict(ga,verb(dan_5,g),meirei).
gdict(ga,verb(sino_1,g),nizen).
gdict(ga,verb(sino_1,g),renyou).
gdict(gere,verb(sino_1,g),katei).
gdict(gero,verb(sino_1,g),meirei).
gdict(geru,verb(sino_1,g),rentai).
gdict(geru,verb(sino_1,g),syuusi).
gdict(gayo,verb(sino_1,g),meirei).
gdict(gi,verb(dan_5,g),renyou).
gdict(gi,verb(kani_1,g),nizen).
gdict(gi,verb(kani_1,g),renyou).
gdict(gire,verb(kani_1,g),katei).
gdict(giro,verb(kani_1,g),meirei).
gdict(giru,verb(kani_1,g),rentai).

```

gdiot(giru, verb(kami_1, g), syuusi).
 gdiot(giyo, verb(kami_1, g), meirei).
 gdiot(gu, verb(dan_5, g), rentai).
 gdiot(gu, verb(dan_5, g), syuusi).
 gdiot(he, verb(simo_1, h), mizen).
 gdiot(he, verb(simo_1, h), renyou).
 gdiot(heze, verb(simo_1, h), katei).
 gdiot(heru, verb(simo_1, h), meirei).
 gdiot(heru, verb(simo_1, h), rentai).
 gdiot(heru, verb(simo_1, h), syuusi).
 gdiot(heyo, verb(simo_1, h), meirei).
 gdiot(hi, verb(kami_1, h), mizen).
 gdiot(hi, verb(kami_1, h), renyou).
 gdiot(hire, verb(kami_1, h), katei).
 gdiot(hiro, verb(kami_1, h), meirei).
 gdiot(hiru, verb(kami_1, h), rentai).
 gdiot(hiru, verb(kami_1, h), syuusi).
 gdiot(hiyo, verb(kami_1, h), meirei).
 gdiot(i, adj, rentai).
 gdiot(i, adj, syuusi).
 gdiot(i, verb(dan_5, g), renyou).
 gdiot(i, verb(dan_5, k), renyou).
 gdiot(i, verb(kami_1, a), mizen).
 gdiot(i, verb(kami_1, a), renyou).
 gdiot(iro, verb(kami_1, a), katei).
 gdiot(iro, verb(kami_1, a), meirei).
 gdiot(iru, verb(kami_1, a), rentai).
 gdiot(iru, verb(kami_1, a), syuusi).
 gdiot(iyo, verb(kami_1, a), meirei).
 gdiot(ka, verb(dan_5, k), mizen).
 gdiot(karo, adj, mizen).
 gdiot(kat, adj, renyou).
 gdiot(ke, verb(dan_5, k), katei).
 gdiot(ke, verb(dan_5, k), meirei).
 gdiot(ke, verb(simo_1, k), mizen).
 gdiot(ke, verb(simo_1, k), renyou).
 gdiot(kere, adj, katei).
 gdiot(kere, verb(simo_1, k), katei).
 gdiot(kero, verb(simo_1, k), meirei).
 gdiot(koru, verb(simo_1, k), rentai).
 gdiot(keru, verb(simo_1, k), syuusi).
 gdiot(keyo, verb(simo_1, k), meirei).
 gdiot(ki, verb(dan_5, k), renyou).
 gdiot(ki, verb(kahen, k), renyou).
 gdiot(ki, verb(kami_1, k), mizen).
 gdiot(ki, verb(kami_1, k), renyou).
 gdiot(kire, verb(kami_1, k), katei).
 gdiot(kiro, verb(kami_1, k), meirei).
 gdiot(kiru, verb(kami_1, k), rentai).
 gdiot(kiru, verb(kami_1, k), syuusi).
 gdiot(kiyo, verb(kami_1, k), meirei).
 gdiot(ko, verb(kahen, k), mizen).
 gdiot(koi, verb(kahen, k), meirei).
 gdiot(ku, adj, renyou).
 gdiot(ku, verb(dan_5, k), rentai).
 gdiot(ku, verb(dan_5, k), syuusi).
 gdiot(kure, verb(kahen, k), katei).

gdiot(kuru, verb(kahan, k), rentai).
 gdiot(kuru, verb(kahan, k), syuusi).
 gdiot(na, verb(dan_5, n), mizen).
 gdiot(na, verb(dan_5, n), katai).
 gdiot(na, verb(dan_5, n), meirei).
 gdiot(na, verb(sino_1, n), mizen).
 gdiot(na, verb(sino_1, n), renyou).
 gdiot(nere, verb(sino_1, n), katai).
 gdiot(nero, verb(sino_1, n), meirei).
 gdiot(neru, verb(sino_1, n), rentai).
 gdiot(neru, verb(sino_1, n), syuusi).
 gdiot(neyo, verb(sino_1, n), meirei).
 gdiot(ni, verb(dan_5, n), renyou).
 gdiot(ni, verb(kami_1, n), mizen).
 gdiot(ni, verb(kami_1, n), renyou).
 gdiot(nire, verb(kami_1, n), katai).
 gdiot(niro, verb(kami_1, n), meirei).
 gdiot(niru, verb(kami_1, n), rentai).
 gdiot(niru, verb(kami_1, n), syuusi).
 gdiot(niyo, verb(kami_1, n), meirei).
 gdiot(nu, verb(dan_5, n), rentai).
 gdiot(nu, verb(dan_5, n), syuusi).
 gdiot(n, verb(dan_5, b), renyou).
 gdiot(n, verb(dan_5, n), renyou).
 gdiot(n, verb(dan_5, n), renyou).
 gdiot(na, adv, rentai).
 gdiot(na, verb(dan_5, n), mizen).
 gdiot(nara, adv, katai).
 gdiot(na, verb(dan_5, n), katai).
 gdiot(na, verb(dan_5, n), meirei).
 gdiot(na, verb(sino_1, n), mizen).
 gdiot(na, verb(sino_1, n), renyou).
 gdiot(nere, verb(sino_1, n), katai).
 gdiot(nero, verb(sino_1, n), meirei).
 gdiot(neru, verb(sino_1, n), rentai).
 gdiot(neru, verb(sino_1, n), syuusi).
 gdiot(neyo, verb(sino_1, n), meirei).
 gdiot(ni, adv, renyou).
 gdiot(ni, verb(dan_5, n), renyou).
 gdiot(ni, verb(kami_1, n), mizen).
 gdiot(ni, verb(kami_1, n), renyou).
 gdiot(niro, verb(kami_1, n), katai).
 gdiot(niro, verb(kami_1, n), meirei).
 gdiot(niru, verb(kami_1, n), rentai).
 gdiot(niru, verb(kami_1, n), syuusi).
 gdiot(niyo, verb(kami_1, n), meirei).
 gdiot(nu, verb(dan_5, n), rentai).
 gdiot(nu, verb(dan_5, n), syuusi).
 gdiot(re, verb(dan_5, r), mizen).
 gdiot(re, verb(dan_5, r), katai).
 gdiot(re, verb(dan_5, r), meirei).
 gdiot(re, verb(sino_1, r), mizen).
 gdiot(re, verb(sino_1, r), renyou).
 gdiot(rere, verb(sino_1, r), katai).
 gdiot(rero, verb(sino_1, r), meirei).
 gdiot(neru, verb(sino_1, r), rentai).
 gdiot(neru, verb(sino_1, r), syuusi).

gdict(reyo, verb(simo_1, r), meirei).
 gdict(ri, verb(dan_5, r), renyou).
 gdict(ri, verb(kami_1, r), mizen).
 gdict(ri, verb(kami_1, r), renyou).
 gdict(rire, verb(kami_1, r), katei).
 gdict(riro, verb(kami_1, r), meirei).
 gdict(riru, verb(kami_1, r), rentai).
 gdict(riru, verb(kami_1, r), syuusi).
 gdict(riyo, verb(kami_1, r), meirei).
 gdict(ru, verb(dan_5, r), rentai).
 gdict(ru, verb(dan_5, r), syuusi).
 gdict(sa, verb(dan_5, s), mizen).
 gdict(sa, verb(sahen, s), mizen).
 gdict(sa, verb(dan_5, s), katei).
 gdict(sa, verb(dan_5, s), meirei).
 gdict(se, verb(sahen, s), mizen).
 gdict(se, verb(simo_1, s), mizen).
 gdict(se, verb(simo_1, s), renyou).
 gdict(sere, verb(simo_1, s), katei).
 gdict(sere, verb(simo_1, s), meirei).
 gdict(seru, verb(simo_1, s), rentai).
 gdict(seru, verb(simo_1, s), syuusi).
 gdict(seyo, verb(sahen, s), meirei).
 gdict(seyo, verb(simo_1, s), meirei).
 gdict(si, verb(dan_5, s), renyou).
 gdict(si, verb(sahen, s), mizen).
 gdict(si, verb(sahen, s), renyou).
 gdict(sirc, verb(sahen, s), meirei).
 gdict(su, verb(dan_5, s), rentai).
 gdict(su, verb(dan_5, s), syuusi).
 gdict(sure, verb(sahen, s), katei).
 gdict(suru, verb(sahen, s), rentai).
 gdict(suru, verb(sahen, s), syuusi).
 gdict(t, verb(dan_5, r), renyou).
 gdict(t, verb(dan_5, t), renyou).
 gdict(t, verb(dan_5, w), renyou).
 gdict(ta, verb(dan_5, t), mizen).
 gdict(te, verb(dan_5, t), katei).
 gdict(te, verb(dan_5, t), meirei).
 gdict(te, verb(simo_1, t), mizen).
 gdict(te, verb(simo_1, t), renyou).
 gdict(tere, verb(simo_1, t), katei).
 gdict(tero, verb(simo_1, t), meirei).
 gdict(teru, verb(simo_1, t), rentai).
 gdict(teru, verb(simo_1, t), syuusi).
 gdict(tayo, verb(simo_1, t), meirei).
 gdict(ti, verb(dan_5, t), renyou).
 gdict(ti, verb(kami_1, t), mizen).
 gdict(ti, verb(kami_1, t), renyou).
 gdict(tire, verb(kami_1, t), katei).
 gdict(tiro, verb(kami_1, t), meirei).
 gdict(tiru, verb(kami_1, t), rentai).
 gdict(tiru, verb(kami_1, t), syuusi).
 gdict(tiyo, verb(kami_1, t), meirei).
 gdict(tu, verb(dan_5, t), rentai).
 gdict(tu, verb(dan_5, t), syuusi).
 gdict(wa, verb(dan_5, w), mizen).

gdiot(ue, verb(fan_5, u), katai).
 gdiot(ue, verb(fan_5, u), meirei).
 gdiot(wi, verb(fan_5, u), renyou).
 gdiot(wu, verb(fan_5, u), rentai).
 gdiot(wu, verb(fan_5, u), syuusi).
 gdiot(za, verb(saken, z), nizen).
 gdiot(ze, verb(saken, z), nizen).
 gdiot(za, verb(sino_1, z), nizen).
 gdiot(ze, verb(sino_1, z), renyou).
 gdiot(zere, verb(sino_1, z), katai).
 gdiot(zero, verb(sino_1, z), meirei).
 gdiot(zeru, verb(sino_1, z), rentai).
 gdiot(zeru, verb(sino_1, z), syuusi).
 gdiot(zezo, verb(saken, z), meirei).
 gdiot(zezo, verb(sino_1, z), meirei).
 gdiot(zi, verb(kani_1, z), nizen).
 gdiot(zi, verb(kani_1, z), renyou).
 gdiot(zi, verb(sanen, z), nizen).
 gdiot(zi, verb(saken, z), renyou).
 gdiot(zire, verb(kani_1, z), katai).
 gdiot(ziro, verb(kani_1, z), meirei).
 gdiot(ziro, verb(saken, z), meirei).
 gdiot(ziru, verb(kani_1, z), rentai).
 gdiot(ziru, verb(kani_1, z), syuusi).
 gdiot(ziyo, verb(kani_1, z), meirei).
 gdiot(zure, verb(saken, z), katai).
 gdiot(zuru, verb(saken, z), rentai).
 gdiot(zuru, verb(saken, z), syuusi).

2.6. 実行例

```
| ?- compile([j_dog,'adict.sorted','hdict.sorted','gdict.sorted']).
j_dog compiled: 1119 words,      1.92 sec.

adict.sorted compiled: 2334 words,    5.75 sec.

hdict.sorted compiled: 2825 words,    4.86 sec.

gdict.sorted compiled: 4345 words,    9.90 sec.

yes
| ?- bunsetu(X,Y,"kaku",[1]).
runtime=5 ms

X = [word(kaku,verb,reg_type(verb(dan_5,k)),rentai)],
Y = word(kaku,verb,reg_type(verb(dan_5,k)),rentai) ;
runtime=103 ms

X = [word(kaku,verb,reg_type(verb(dan_5,k)),syuusi)],
Y = word(kaku,verb,reg_type(verb(dan_5,k)),syuusi) ;

no
| ?- bunsetu(X,Y,"kakitakunai",[]).
runtime=57 ms

X = [word(kaku,verb,reg_type(verb(dan_5,k)),renyou),word(tai,auxv,reg_type(adj),
renyou),word(nai,auxv,reg_type(adj),rentai)],
Y = word(nai,auxv,reg_type(adj),rentai) ;
runtime=142 ms

X = [word(kaku,verb,reg_type(verb(dan_5,k)),renyou),word(tai,auxv,reg_type(adj),
renyou),word(nai,auxv,reg_type(adj),syuusi)],
Y = word(nai,auxv,reg_type(adj),syuusi) ;

no
| ?- bunsetu(X,Y,"kakaseraretakunakattayocuda",[]).
runtime=324 ms

X = [word(kaku,verb,reg_type(verb(dan_5,k)),mizen),word(seru,auxv,reg_type(verb(
simo_1,s)),mizen),word(rareru,auxv,reg_type(verb(simo_1,r)),renyou),word(tai,aux
v,reg_type(adj),renyou),word(nai,auxv,reg_type(adj),renyou),word(ta,auxv,irreg_t
ype([(mizen,[114,111]),(syuusi,[1]),(rentai,[1]),(katei,[114,97])]),rentai),word(y
ouda,auxv,irreg_type([(mizen,[100,97,114,111]),(renyou,[100,97,115]),(renyou,[10
0,101]),(renyou,[110,105]),(syuusi,[100,97]),(rentai,[110,97]),(katei,[110,97,11
4,97])]),syuusi)],
Y = word(youda,auxv,irreg_type([(mizen,[100,97,114,111]),(renyou,[100,97,115]),(
renyou,[100,101]),(renyou,[110,105]),(syuusi,[100,97]),(rentai,[110,97]),(katei,
[110,97,114,97])]),syuusi) ;

no
```