TR-IT-0015

C言語による双方向CHART構文解析法(II)――プログラム解説書――

Bidirectional Chart Parser: C language version (II) Manual

ローケン・キム キュンホ セリグマン マーク 北川 美宏*

Kyung-ho LOKEN-KIM Mark SELIGMAN Yosihiro KITAGAWA*

1993 .9

概要

CHART解析法は、構文解析法の中でも柔軟性のある解析法である。下降型あるいは上昇型での解析が選択できる上、それぞれに対して、縦型あるいは横型の探索方式が選択できる。しかもそれぞれを混在させることも可能である。今回は、簡単な文脈自由文法上で動く双方向CHART構文解析法をC言語により作成したので報告する。次葉から、第２章では実行モジュールの作成方法を、第３章では実行方法と実行時のオプションを、第４章ではプログラムの内容を、第５章ではテストに用いた文法ファイルを、それぞれ簡単に説明する。

ATR Interpreting Telecommunications Research Laboratories

ATR 音声翻訳通信研究所
*国際電気通信基礎技術研究所

目次

第１章　まえがき

構文解析において、解析対象となる文中、最も信頼度の高い要素(Anchor Symbol)から解析を始め、右方向だけでなく双方向(Bidirectional)に解析を進める方法(Island Driven)が提案されている。
ここでは、構文解析アルゴリズムの中でも柔軟性のあるアルゴリズムであるCHARTアルゴリズムを応用し、Island Drivenな解析を行う双方向型CHARTパーザーを試作した。なお今回は問題を簡単にするため、信頼度をランダムに付与したワードを最小の処理単位とした。
以降、第２章では実行モジュールの作成方法を、第３章では実行方法と実行時のオプションを、第４章ではプログラムの内容を、第５章ではテストに用いた文法ファイルを、それぞれ簡単に説明する。そして最後に、第６章では今回のプロトタイプシステムの評価と今後の課題を簡単に述べる。

第２章　実行モジュールの作成方法

実行モジュールの作成には、以下のファイルが必要である。

```
chart.c          chart_util.c     read_datafile.c          score.c
test_print.c     chart.h          chart_util.h             Makefile
```

これらのファイルを同じディレクトリに入れmakeを実行すると、実行モジュール"chart"ができる。

[実行例]

```
$ make
cc -g -c chart.c
cc -g -c chart_util.c
cc -g -c read_datafile.c
cc -g -c score.c
cc -g -c test_print.c
cc -g -o chart chart.o  chart_util.o  read_datafile.o  score.o  test_print.o
```

第３章　実行方法

3.1　起動時オプション

```
-g FILE-NAME     :文法ファイルの指定
-s FILE_NAME     :入力文ファイルの指定
-p               :解析結果表示の際、各エッジのScoreも表示
```

3.2　起動例

文法ファイル"grammer5.dat"と入力文ファイル"sentence.dat"を読み込んで解析を行う。解析結果の表示には各エッジのScoreも表示する。

```
$ chart -g grammer5.dat -s sentence.dat -p
```

3.3　実行結果例

文法ファイル:"grammer5.dat"

```
RULE S -> NP VP;
RULE VP -> IV;
RULE VP -> IV PP;
RULE VP -> TV NP;
RULE VP -> TV NP PP;
RULE VP -> TV NP VP;
RULE NP -> Det N;
RULE NP -> Det N PP;
RULE PP -> P NP;
WORD the = Det;
WORD her = Det;
WORD her = NP;
WORD they = NP;
WORD nurses = NP;
WORD nurses = N;
WORD book = N;
WORD travel = N;
WORD report = N;
WORD hear = IV;
```

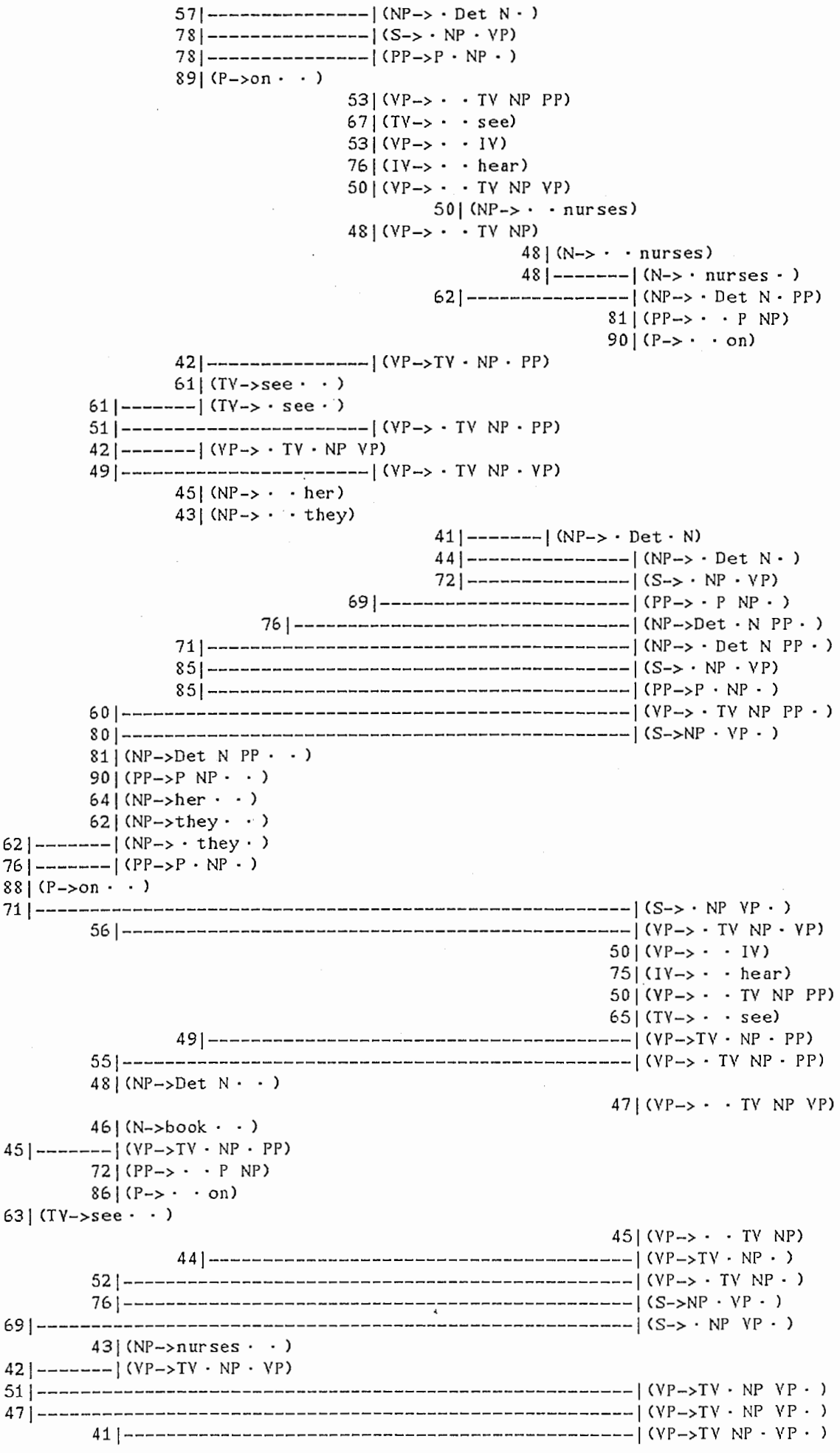
```
WORD hear = TV;
WORD see = TV;
WORD on = P;
```

入力文ファイル:"sentence.dat"

```
They see the book on the nurses.
```

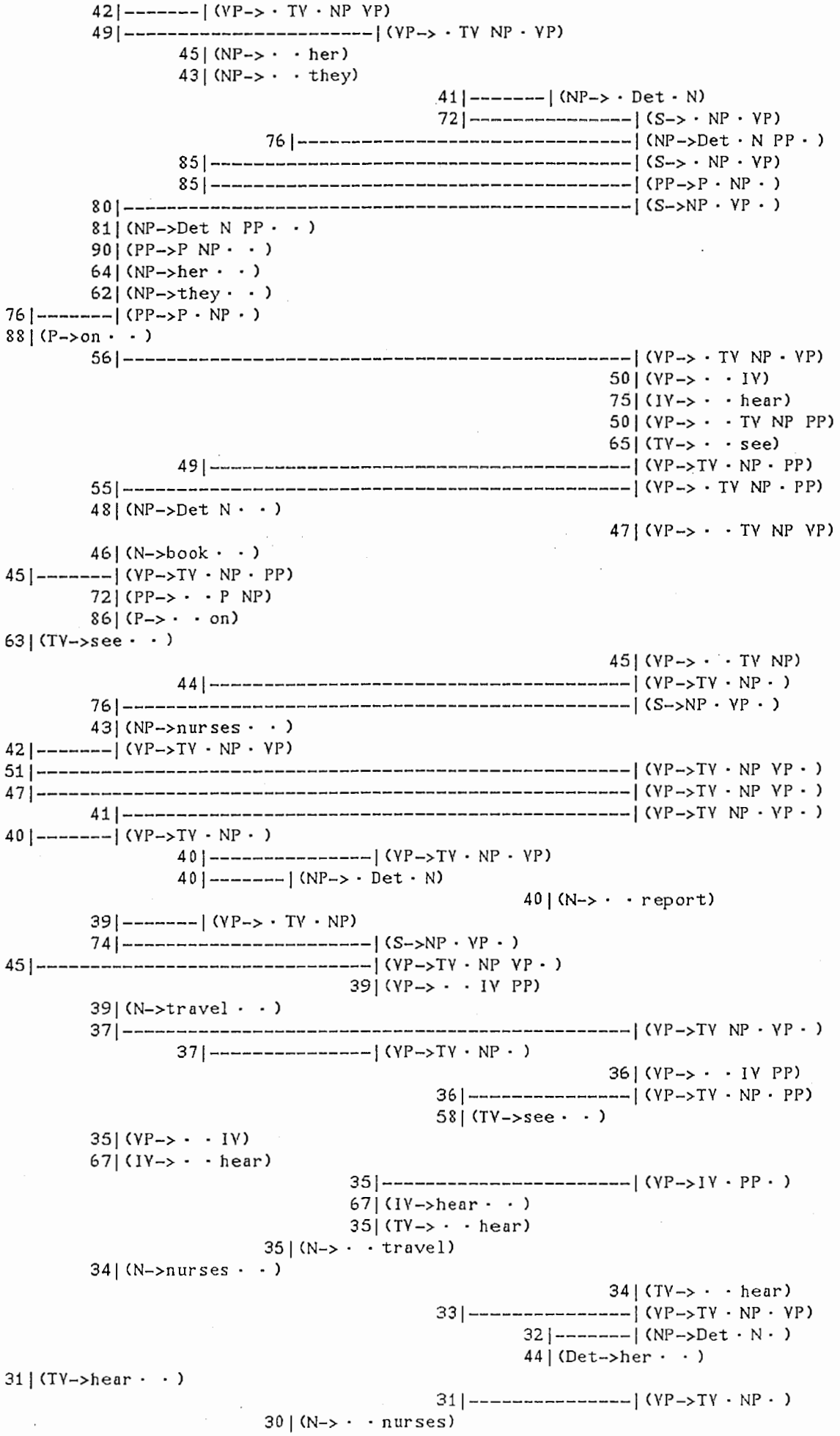
実行:

```
$ chart -g grammer5.dat -s sentence.dat -p

100   S -> NP VP                  /*---------------------------------------*/
 28   VP -> IV                    /* 構文規則にもScoreをランダムに付与     */
  1   VP -> IV PP                 /* ただし、左辺が同じ規則のScore合計が   */
 18   VP -> TV NP                 /*   約100となるよう調整                 */
 28   VP -> TV NP PP              /*---------------------------------------*/
 23   VP -> TV NP VP
 17   NP -> Det N
 82   NP -> Det N PP
100   PP -> P NP
 43   Det -> the
 56   Det -> her
 48   NP -> her
 44   NP -> they
  7   NP -> nurses
 20   N -> nurses
 44   N -> book
 30   N -> travel
  4   N -> report
100   IV -> hear
 18   TV -> hear
 81   TV -> see
100   P -> on
                                  /*---------------------------------------*/
                                  /* 入力文およびIslandを表示              */
                                  /*---------------------------------------*/
--- SENTENCE ---
they(14) see(11) the(73) book(86) on(39) the(45) nurses(77) .
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
/*** ISLAND ************************************************/
                          86|-------|(N-> · book · )
/***********************************************  ISLAND ***/

                                  /*---------------------------------------*/
                                  /* CHARTに加えたエッジを逐次表示         */
                                  /*---------------------------------------*/
                          86|-------|(N-> · book · )
                          84|-------|(NP->Det · N · PP)
                                  92|(PP-> · · P NP)
                                  96|(P-> · · on)
                                  96|-------|(P-> · on · )
                                  94|-------|(PP-> · P · NP)
                                          88|(NP-> · · Det N PP)
                                          72|(Det-> · · her)
                                          71|(NP-> · · her)
                          70|(Det->her · · )
                                          69|(NP-> · · they)
                                          65|(Det-> · · the)
                                          65|-------|(Det-> · the · )
                                          76|-------|(NP-> · Det · N PP)
                          63|(Det->the · · )
                  63|-------|(Det-> · the · )
                  73|---------------|(NP-> · Det N · PP)
                                                  60|(N-> · · book)
                                          55|(NP-> · · Det N)
                                                  53|(N-> · · travel)
                          51|-------|(NP->Det · N · )
```


57|(NP-> · Det N ·)
78|(S-> · NP · VP)
78|(PP->P · NP ·)
89|(P->on · ·)
53|(VP-> · · TV NP PP)
67|(TV-> · · see)
53|(VP-> · · IV)
76|(IV-> · · hear)
50|(VP-> · · TV NP VP)
50|(NP-> · · nurses)
48|(VP-> · · TV NP)
48|(N-> · · nurses)
48|(N-> · nurses ·)
62|(NP-> · Det N · PP)
81|(PP-> · · P NP)
90|(P-> · · on)
42|(VP->TV · NP · PP)
61|(TV->see · ·)
61|(TV-> · see ·)
51|(VP-> · TV NP · PP)
42|(VP-> · TV · NP VP)
49|(VP-> · TV NP · VP)
45|(NP-> · · her)
43|(NP-> · · they)
41|(NP-> · Det · N)
44|(NP-> · Det N ·)
72|(S-> · NP · VP)
69|(PP-> · P NP ·)
76|(NP->Det · N PP ·)
71|(NP-> · Det N PP ·)
85|(S-> · NP · VP)
85|(PP->P · NP ·)
60|(VP-> · TV NP PP ·)
80|(S->NP · VP ·)
81|(NP->Det N PP · ·)
90|(PP->P NP · ·)
64|(NP->her · ·)
62|(NP->they · ·)
62|(NP-> · they ·)
76|(PP->P · NP ·)
88|(P->on · ·)
71|(S-> · NP VP ·)
56|(VP-> · TV NP · VP)
50|(VP-> · · IV)
75|(IV-> · · hear)
50|(VP-> · · TV NP PP)
65|(TV-> · · see)
49|(VP->TV · NP · PP)
55|(VP-> · TV NP · PP)
48|(NP->Det N · ·)
47|(VP-> · · TV NP VP)
46|(N->book · ·)
45|(VP->TV · NP · PP)
72|(PP-> · · P NP)
86|(P-> · · on)
63|(TV->see · ·)
45|(VP-> · · TV NP)
44|(VP->TV · NP ·)
52|(VP-> · TV NP ·)
76|(S->NP · VP ·)
69|(S-> · NP VP ·)
43|(NP->nurses · ·)
42|(VP->TV · NP · VP)
51|(VP->TV · NP VP ·)
47|(VP->TV · NP VP ·)
41|(VP->TV NP · VP ·)

```
 40|-------|(VP->TV · NP · )
                 40|----------------|(VP->TV · NP · VP)
                 40|-------|(NP-> · Det · N)
                                                  40|(N-> · · report)
         39|-------|(VP-> · TV · NP)
         48|-----------------------|(VP-> · TV NP · )
         74|-----------------------|(S->NP · VP · )
 68|------------------------------|(S-> · NP VP · )
 45|------------------------------|(VP->TV · NP VP · )
                              39|(VP-> · · IV PP)
         39|(N->travel · · )
         37|-----------------------------------------------|(VP->TV NP · VP · )
                 37|----------------|(VP->TV · NP · )
                                                  36|(VP-> · · IV PP)
                                        36|----------------|(VP->TV · NP · PP)
                                        58|(TV->see · · )
         35|(VP-> · · IV)
         67|(IV-> · · hear)
                              35|-----------------------|(VP->IV · PP · )
                              67|(IV->hear · · )
                              35|(TV-> · · hear)
                        35|(N-> · · travel)
         34|(N->nurses · · )
                                                  34|(TV-> · · hear)
                                        33|----------------|(VP->TV · NP · VP)
                                                32|-------|(NP->Det · N · )
                                                44|(Det->her · · )
 31|(TV->hear · · )
                                        31|----------------|(VP->TV · NP · )
                        30|(N-> · · nurses)
                 30|(TV->hear · · )
                                        27|(TV->hear · · )
         26|(N->report · · )
                 24|(NP-> · · nurses)
                        22|(N-> · · report)
         21|(VP-> · · IV PP)


                              /*---------------------------------------*/
                              /* 解析結果を表示                        */
                              /*---------------------------------------*/
[SCORE]
     they     see      the      book     on       the      nurses
   0------ 1------ 2------ 3------ 4------ 5------ 6------ 7
       14       11       73       86       39       45       77

>>>>>>> RESULTs <<<<<<<
 71|-----------------------------------------------------|(S-> · NP VP · )
 62|-------|(NP-> · they · )
         60|-----------------------------------------------|(VP-> · TV NP PP · )
         61|-------|(TV-> · see · )
                 57|----------------|(NP-> · Det N · )
                 63|-------|(Det-> · the · )
                         86|-------|(N-> · book · )
                                 69|-----------------------|(PP-> · P NP · )
                                 96|-------|(P-> · on · )
                                         44|----------------|(NP-> · Det N · )
                                         65|-------|(Det-> · the · )
                                                 48|-------|(N-> · nurses · )


 69|-----------------------------------------------------|(S-> · NP VP · )
 62|-------|(NP-> · they · )
         52|-----------------------------------------------|(VP-> · TV NP · )
         61|-------|(TV-> · see · )
                 71|-------------------------------------|(NP-> · Det N PP · )
```

```
                    63|-------|(Det-> · the · )
                            86|-------|(N-> · book · )
                                    69|-----------------------|(PP-> · P NP · )
                                    96|-------|(P-> · on · )
                                            44|---------------|(NP-> · Det N · )
                                            65|-------|(Det-> · the · )
                                                    48|-------|(N-> · nurses · )


>>> INACTIVE_EDGE <<<
                            86|-------|(N-> · book · )
                                    96|-------|(P-> · on · )
                                            65|-------|(Det-> · the · )
                    63|-------|(Det-> · the · )
                    57|---------------|(NP-> · Det N · )
                                                    48|-------|(N-> · nurses · )
            61|-------|(TV-> · see · )
                                            44|---------------|(NP-> · Det N · )
                                    69|-----------------------|(PP-> · P NP · )
                    71|---------------------------------------|(NP-> · Det N PP · )
            60|---------------------------------------------------|(VP-> · TV NP PP · )
    62|-------|(NP-> · they · )
    71|-------------------------------------------------------------|(S-> · NP VP · )
            52|-----------------------------------------------|(VP-> · TV NP · )
    69|-------------------------------------------------------|(S-> · NP VP · )
            48|-----------------------|(VP-> · TV NP · )
    68|-------------------------------|(S-> · NP VP · )


>>> ACTIVE_EDGE <<<
                            84|-------|(NP->Det · N · PP)
                                    92|(PP-> · · P NP)
                                    96|(P-> · · on)
                                    94|-------|(PP-> · P · NP)
                                            88|(NP-> · · Det N PP)
                                            72|(Det-> · · her)
                                            71|(NP-> · · her)
                            70|(Det->her · · )
                                            69|(NP-> · · they)
                                            65|(Det-> · · the)
                                            76|-------|(NP-> · Det · N PP)
                            63|(Det->the · · )
                    73|---------------|(NP-> · Det N · PP)
                                                    60|(N-> · · book)
                                            55|(NP-> · · Det N)
                                                    53|(N-> · · travel)
                            51|-------|(NP->Det · N · )
                    78|---------------|(S-> · NP · VP)
                    78|---------------|(PP->P · NP · )
                    89|(P->on · · )
                                    53|(VP-> · · TV NP PP)
                                    67|(TV-> · · see)
                                    53|(VP-> · · IV)
                                    76|(IV-> · · hear)
                                    50|(VP-> · · TV NP VP)
                                            50|(NP-> · · nurses)
                                    48|(VP-> · · TV NP)
                                                    48|(N-> · · nurses)
                                            62|---------------|(NP-> · Det N · PP)
                                                            81|(PP-> · · P NP)
                                                            90|(P-> · · on)
                    42|---------------|(VP->TV · NP · PP)
                    61|(TV->see · · )
            51|-----------------------|(VP-> · TV NP · PP)
```


42|-------|(VP-> · TV · NP VP)
49|----------------------|(VP-> · TV NP · VP)
45|(NP-> · · her)
43|(NP-> · · they)
41|-------|(NP-> · Det · N)
72|----------------|(S-> · NP · VP)
76|--------------------------------|(NP->Det · N PP ·)
85|-----------------------------------|(S-> · NP · VP)
85|-----------------------------------|(PP->P · NP ·)
80|---|(S->NP · VP ·)
81|(NP->Det N PP · ·)
90|(PP->P NP · ·)
64|(NP->her · ·)
62|(NP->they · ·)
76|-------|(PP->P · NP ·)
88|(P->on · ·)
56|---|(VP-> · TV NP · VP)
50|(VP-> · · IV)
75|(IV-> · · hear)
50|(VP-> · · TV NP PP)
65|(TV-> · · see)
49|-----------------------------------|(VP->TV · NP · PP)
55|---|(VP-> · TV NP · PP)
48|(NP->Det N · ·)
47|(VP-> · · TV NP VP)
46|(N->book · ·)
45|-------|(VP->TV · NP · PP)
72|(PP-> · · P NP)
86|(P-> · · on)
63|(TV->see · ·)
45|(VP-> · · TV NP)
44|-----------------------------------|(VP->TV · NP ·)
76|---|(S->NP · VP ·)
43|(NP->nurses · ·)
42|-------|(VP->TV · NP · VP)
51|---|(VP->TV · NP VP ·)
47|---|(VP->TV · NP VP ·)
41|---|(VP->TV NP · VP ·)
40|-------|(VP->TV · NP ·)
40|----------------|(VP->TV · NP · VP)
40|-------|(NP-> · Det · N)
40|(N-> · · report)
39|-------|(VP-> · TV · NP)
74|----------------------|(S->NP · VP ·)
45|------------------------------|(VP->TV · NP VP ·)
39|(VP-> · · IV PP)
39|(N->travel · ·)
37|---|(VP->TV NP · VP ·)
37|----------------|(VP->TV · NP ·)
36|(VP-> · · IV PP)
36|----------------|(VP->TV · NP · PP)
58|(TV->see · ·)
35|(VP-> · · IV)
67|(IV-> · · hear)
35|----------------------|(VP->IV · PP ·)
67|(IV->hear · ·)
35|(TV-> · · hear)
35|(N-> · · travel)
34|(N->nurses · ·)
34|(TV-> · · hear)
33|----------------|(VP->TV · NP · VP)
32|-------|(NP->Det · N ·)
44|(Det->her · ·)
31|(TV->hear · ·)
31|----------------|(VP->TV · NP ·)
30|(N-> · · nurses)

```
                    30|(TV->hear ・ ・)
                                                27|(TV->hear ・ ・)
              26|(N->report ・ ・)
                    24|(NP->・ ・nurses)
                              22|(N->・ ・report)
              21|(VP->・ ・IV PP)

/*-------------------------------------------------------------------------*/
/* エッジの表示について:                                                   */
/*                                                                         */
/*          they    see     the     book    on      the     nurses         */
/*         0------ 1------ 2------ 3------ 4------ 5------ 6------ 7       */
/*                                                                         */
/*                                                                         */
/*                              84|-------|(NP->Det ・ N ・ PP)            */
/*      このエッジの内容は、                                               */
/*              Score           84                                         */
/*              Label           NP                                         */
/*              Rule            NP→Det N PP                                */
/*              FromPosition    1                                          */
/*              ToPosition      2                                          */
/*              StartVertex     3                                          */
/*              FinishVertex    4                                          */
/*      となる。すなわち、|-------|は、上の文に対応してエッジがどこからどこまで */
/*      のびているかを表している。その左の数字は、各エッジのScoreである。右の構 */
/*      文規則は、各エッジのルールであり、左・と右・の間にある標識が解析できた  */
/*      部分である。エッジの解析が進むと、左・は左方向へ、右・は右方向へ移動し、 */
/*      それぞれが左端、右端まで到達すれば、当該エッジの解析が終了する。       */
/*                                                                         */
/*-------------------------------------------------------------------------*/
```

第4章　プログラム

4.1　ファイル構成

各ファイルには、おおまかに以下の内容の関数、データ定義等が記述してある。

"chart.c"	:main()関数の他、CHARTアルゴリズムを実行する関数群
"chart_util.c"	:メモリ管理、ストリング管理の関数群
"read_datafile.c"	:文法ファイル、入力文ファイルの読み込み関数
"score.c"	:乱数を利用し、テスト用の信頼度を返す関数
"test_print.c"	:デバッグ表示用の関数群
"chart.h"	:CHART用のデータ構造他
"chart_util.h"	:標準データタイプのtypedef他

4.2　データ構造

[EDGE]

```
struct _cpEDGE {
        LONG            Score;          /* 信頼度 */
        cpEDGETYPE      Type;           /* エッジ種別 */
        SHORT           StartVertex;    /* エッジの出発点 */
        SHORT           FinishVertex;   /* エッジの到着点 */
        SHORT           FromPosition;   /* 解析済標識列先頭位置 */
                                        /* ( A->・B C・D:FromPosition=0) */
        SHORT           ToPosition;     /* 解析済標識列末尾位置 */
                                        /* ( A->・B C・D:ToPosition=2) */
        cpCATEGORY      Label;          /* ラベル */
        cpPRULE         pRule;          /* 構文規則 */
        cpPDAUGHTER_L   pDaughter;      /* 娘エッジリスト */
};

enum _cpEDGETYPE {
```

```
        TERMINAL_SYMBOL_EDGE,           /* 前終端標識エッジ */
        INACTIVE_EDGE,                  /* 非活性エッジ */
        ACTIVE_EDGE                     /* 活性エッジ */
};

enum _cpCATEGORY {
        CATEGORY_S,                     /* 文 */
        CATEGORY_NP,                    /* 名詞句 */
        CATEGORY_VP,                    /* 動詞句 */
        CATEGORY_PP,                    /* 前置詞句 */
        CATEGORY_N,                     /* 名詞 */
        CATEGORY_IV,                    /* 自動詞 */
        CATEGORY_TV,                    /* 他動詞 */
        CATEGORY_P,                     /* 前置詞 */
        CATEGORY_D,                     /* 冠詞 */
        CATEGORY_NIL,                   /* 該当品詞無し */
        CATEGORY_NUM
};

struct _cpDAUGHTER_L {
        cpPEDGE         pEdge;          /* 娘エッジ */
        cpPDAUGHTER_L   pNext;
};

struct _cpEDGE_L {
        cpPEDGE         pEdge;          /* cpEDGE構造体へのポインタ */
        cpPEDGE_L       pNext;
};
```

[CHART]

```
struct _cpCHART {
        cpPEDGE         pEdge;          /* エッジ */
        cpPCHART        pNext;
};

struct _cpVERTEX {
        cpPEDGE_L       pOutGoingEdgeL; /* 出発エッジリスト */
        cpPEDGE_L       pInComingEdgeL; /* 到着エッジリスト */
};
```

[AGENDA]

```
struct _cpAGENDA {
        cpPEDGE         pEdge;          /* エッジ */
        cpPAGENDA       pPre;           /* 直近Score高AGENDA */
        cpPAGENDA       pNext;          /* 直近Score低AGENDA */
};
```

以上の構造体により、EDGEおよびCHART、AGENDAを管理する。CHARTとAGENDAはともにエッジの集合体であり、実体はEDGEへのポインタリストである。ただし、CHARTでは時系列順(チャートへエッジが加えられた順)にリストを生成するのに対し、AGENDAではエッジのScore順(降順)にソートしたリストを生成する。また初期化時に、チャートの各VERTEXを管理する為のVERTEXテーブルを生成し、エッジの結合など、解析の効率化を図る。

それぞれのリストおよびテーブルは、グローバル変数pChartTop<->pChartBottom、pAgendaTop<->pAgendaBottom、VertexTableにより管理する。

[辞書]

```
struct _cpRULE {
        cpRULETYPE      Type;           /* ルール種別 */
        cpCATEGORY      LhsCategory;    /* 標識(:文法ルール左辺) */
        union _Rhs {
                        cpCATEGORY      Category[RHS_MAX_NUM];
                        PCHAR           pString;
        }               Rhs;            /* 標識群or単語文字列 */
                                        /*   (:文法ルール右辺) */
```

```
        USHORT          RhsCategoryNum; /* 右辺標識数 */
        LONG            Score;          /* ルール信頼度:テスト的に使用 */
        cpPRULE         pNext;
};

enum _cpRULETYPE {
        TERMINAL_SYMBOL,                /* 前終端記号-構文規則 */
        NON_TERMINAL_SYMBOL,            /* 非前終端記号-構文規則 */
};
```

以上の構造体により、文法(句構造規則)および単語を管理する。なお便宜上、句構造規則において、RHS_MAX_NUM(=3)により右辺の標識数の上限を設定しているが、増減させることに問題はない。
句構造規則リストはグローバル変数pFirstRuleにより管理する。

[文]

```
struct _cpSENTENCE {
        PCHAR           pString;        /* 単語文字列 */
        LONG            Score;          /* 信頼度 */
        cpPSENTENCE     pNext;          /* =NULL で文末 */
};
```

以上の構造体により、解析対象の文を管理する。入力文読み込み関数が、入力文ファイルから1文を読み込み、単語単位でリスト構造に展開する。リストはグローバル変数pSentenceTopにより管理する。

```
/*-------------------------------------------------------------------------*/
/* Scoreについて(以降の混乱を回避するための蛇足):                          */
/*                                                                         */
/*      cpSENTENCE、cpEDGE、cpRULE構造体のそれぞれにメンバScoreがある。この */
/*    Scoreは、コメントにもあるとおり、各構造体の信頼度を意味するものである。 */
/*    今回のプログラムでは、この各Scoreをすべて同列に扱い、極々簡単な計算をも */
/*    とに新しい構造体のScoreを算出している。しかし本来、単語、エッジ、構文規 */
/*    則のそれぞれは、まったく性質の違う信頼度を持つはずのものであり、その点に*/
/*    おいて、ここでのScoreは妥当性を欠いたものである。                   */
/*      しかしながら「Scoreを用いて解析の制御を行うこと」は、実現してあるので */
/*    将来、信頼度の扱い方が明確になれば、それに対応することは十分可能である。*/
/*                                                                         */
/*-------------------------------------------------------------------------*/
```

4.3 処理概要

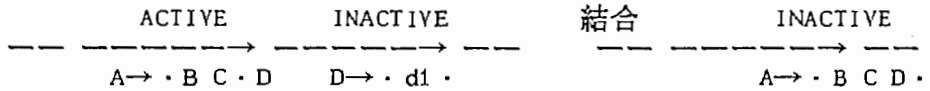
データエリアとしてAGENDAとCHARTを使用する。どちらもエッジの集合であるが、特にAGENDAは、エッジのScore(信頼度)によりソートされたエッジの集合となる。
処理は、「AGENDAから取り出したエッジをCHARTに加える」、「加えたエッジにより新たに生成されるエッジをAGENDAに入れる(詳細は後述)」の繰り返しである。この繰り返しにより解析を進める。

<処理の流れ>

Scoreの高いワードを選び(例えば上位3つまで)、このワードをINACTIVEエッジとして生成し、AGENDAへ入れる。
で生成したINACTIVEエッジをIslandと見立て、これを徐々に大きくしていくことにより解析を進める。解析は、AGENDAからScoreの最も高いエッジを取り出しCHARTに加え、加えたエッジから後述の「エッジ生成の手順」に従って新しいエッジを生成することで進められる。新たに生成されたエッジは、すぐCHARTには加えず、一旦AGENDAに入れる。なおこのときに、新エッジと同じかさらに解析が進んだ状態のエッジが、すでにCHARTもしくはAGENDAに存在する場合、この新エッジはAGENDAへ入れない。以降、この処理を繰り返す。
AGENDAにエッジが無くなれば処理を終了する。

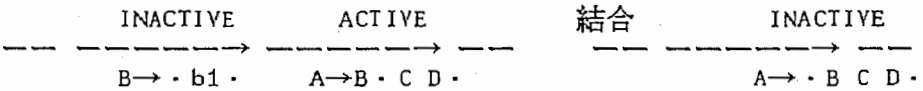
AGENDA→CHART:エッジ生成の手順
1.AGENDAから取り出したエッジがINACTIVEエッジの場合
  ACTIVE_EDGEとの結合(基本ルール)
    左ACTIVE-右INACTIVE(=左 右)

結合条件:
・ACTIVEエッジのFinishVertex=INACTIVEエッジのStartVertex
・ACTIVEエッジのRuleのRhsCategoryNum>ACTIVEエッジのToPosition
・INACTIVEエッジのLabel=ACTIVEエッジのRule.Rhs.Category[ToPosition]
サンプル:



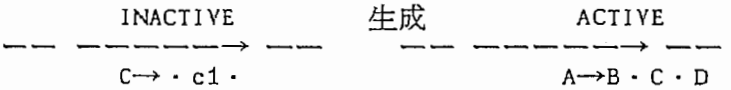
Label	A	D	A
Rule	A→B C D	D→d1	A→B C D
FromPosition	0	0	0
ToPosition	2	1	3
	(Rule.Rhs.Category[ToPosition]=D)		
StartVertex	3	4	3
FinishVertex	4	5	5

左INACTIVE-右ACTIVE(=左 右)
結合条件:
・INACTIVEエッジのFinishVertex=ACTIVEエッジのStartVertex
・ACTIVEエッジのFromPosition>0
・INACTIVEエッジのLabel=ACTIVEエッジのRule.Rhs.Category[FromPosition-1]
サンプル:



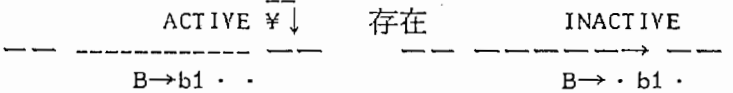
Label	B	A	A
Rule	B→b1	A→B C D	A→B C D
FromPosition	0	1	0
		(Rule.Rhs.Category[FromPosition-1]=B)	
ToPosition	1	3	3
StartVertex	3	4	3
FinishVertex	4	5	5

INACTIVEエッジと構文規則から新たなACTIVEエッジを生成(ボトムアップ)
生成条件:
・INACTIVEエッジのLabel=構文規則右辺のいずれかの標識
サンプル:



Label	C	A
Rule	C→c1	A→B C D
FromPosition	0	1
ToPosition	1	2
StartVertex	3	3
FinishVertex	4	4

2.AGENDAから取り出したエッジがACTIVEエッジの場合
ACTIVEエッジのRuleがTERMINAL_SYMBOLルールの場合は次の処理のみを行い、 、 は行わない。
RuleのStringと一致するワードが、ACTIVEエッジのStartVertex-FinishVertexのあたりに存在するかをチェックする。一致するワードが存在すれば、ACTIVEエッジをINACTIVEエッジとする。
検索位置:
・ACTIVEエッジのFromPosition>0ならStartVertexを-1する
・ACTIVEエッジのToPosition==0ならFinishVertexを+1する
サンプル:



Label	B	B
Rule	B→b1	B→b1
FromPosition	1	0
ToPosition	1	1

StartVertex	3	2
FinishVertex	3	3

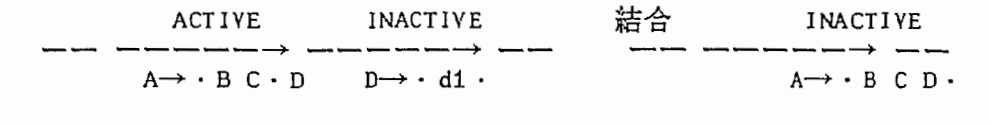
INACTIVE_EDGEとの結合(基本ルール):上記INACTIVEエッジを取り出した場合と同様

左ACTIVE-右INACTIVE(=左 右)

結合条件:

- ACTIVEエッジのFinishVertex=INACTIVEエッジのStartVertex
- ACTIVEエッジのRuleのRhsCategoryNum>ACTIVEエッジのToPosition
- INACTIVEエッジのLabel=ACTIVEエッジのRule.Rhs.Category[ToPosition]

サンプル:



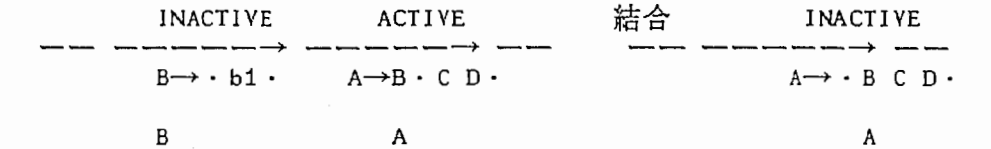
Label	A	D	A
Rule	A→B C D	D→d1	A→B C D
FromPosition	0	0	0
ToPosition	2	1	3
	(Rule.Rhs.Category[ToPosition]=D)		
StartVertex	3	4	3
FinishVertex	4	5	5

左INACTIVE-右ACTIVE(=左 右)

結合条件:

- INACTIVEエッジのFinishVertex=ACTIVEエッジのStartVertex
- ACTIVEエッジのFromPosition>0
- INACTIVEエッジのLabel=ACTIVEエッジのRule.Rhs.Category[FromPosition-1]

サンプル:



Label	B	A	A
Rule	B→b1	A→B C D	A→B C D
FromPosition	0	1	0
		(Rule.Rhs.Category[FromPosition-1]=B)	
ToPosition	1	3	3
StartVertex	3	4	3
FinishVertex	4	5	5

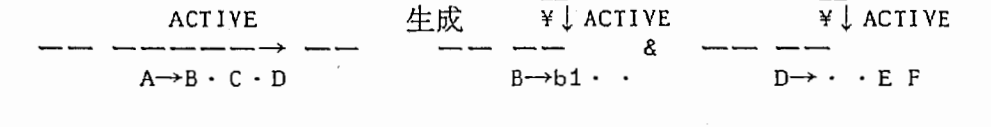
ACTIVEエッジと構文規則から新たなACTIVEエッジを生成(トップダウン)

生成条件:

- ACTIVEエッジの未解析Rule.Rhs.Category=構文規則左辺の標識

(Rule.Rhs.Category[FromPosition-1(:FromPosition>0)]
or Rule.Rhs.Category[ToPosition(:ToPosition<Rule.RhsCategoryNum)])

サンプル:



Label	A	B	D
Rule	A→B C D	B→b1	D→E F
FromPosition	1	1	0
ToPosition	2	1	0
StartVertex	3	3	4
FinishVertex	4	3	4

4.4 関数

「付録A 関数概要」を参照。

第5章 句構造規則および単語辞書、解析対象文

句構造規則および単語辞書は、次の規則に従って記述する。記述ファイルは、関数

ReadGrammarFile()により起動時に読み込まれ、リスト構造に展開される。読み込みファイル名は、オプション-g ファイル名 で指定する。

[句構造規則]

```
句構造規則      :RULE 標識 -> 標識列;
標識            :S | NP | VP | PP | N | IV | TV | P | Det
標識列          :標識 [標識列]

単語            :WORD 文字列 = 標識;
文字列          :(任意の単語を表す文字列)
```

[例]

```
RULE S -> NP VP;
RULE VP -> TV NP PP;
RULE NP -> Det N;
RULE PP -> P NP;
WORD they = NP;
WORD see = TV;
WORD on = P;
WORD the = Det;
WORD book = N;
```

解析対象文は、次の規則に従って記述する。記述ファイルは、関数ReadSentenceFile()により起動時に読み込まれ、リスト構造に展開される。読み込みファイル名は、オプション-s ファイル名 で指定する。なお、読み込んだ文字列は、大文字小文字の区別なく、以降すべて小文字として扱う。

[文]

```
文              :単語列.
単語列          :文字列 [単語列]
文字列          :(任意の単語を表す文字列)
```

[例]

```
They see the book on the nurses.
```

第6章 おわりに

6.1 評価

文の左端(文頭)から右方向(文末方向)へ構文解析を進める従来の方法では、文頭付近に現れるSymbolの信頼度が低い場合に、冗長な処理を誘発したり、間違った解を導いてしまうといった問題が発生する。今回は、この問題を解決するために、文中の信頼度の高いSymbolから解析を始め、右方向にも左方向にも解析を進めることができる「双方向型CHARTパーザー」を試作した。

構文規則、例文ともに最少のデータ量でしかテストを行っていないので、正答率などの定量的な評価はできていないが、問題なく動作し、解析解も正しく導いている。

特に、解析の開始点となる信頼度の高いSymbolをIsland(島)に見立て、徐々にそのIslandを大きくしていくIsland Drivenな解析方法は、適当な構文規則から次に現れるSymbolを予測することができ、将来、音声認識システムとの連携において効果を発揮するであろう。

また、音声認識システムとの連結を想定し、ポーズユニット(:話中、ポーズにより区切られたユニット)毎の解析に対応する機能も試作した(ただし、プログラム説明の部分では特に触れていない)。今回試作したプログラムでは、ひとつのポーズユニットの解析が終了したとき、その結果すなわちCHARTを残しておき、次のポーズユニットが入力されたときに、そのCHARTとあわせて解析を続行する。なお、ポーズユニット毎でIslandは生成するが、CHARTはひとつのものを終始使用する。(本プログラムでは、ポーズマーカーの代わりに入力文中の任意の位置に'.'ピリオドを挿入することで、この機能を確認できる。)

ポーズユニット毎の処理が可能となることで、将来、音声認識システムとの連携処理においてポーズの間にそこまでのポーズユニットの解析を行えれば、見かけ上リアルタイムな処理を実現できる。また、解析側から認識側への信頼度等のフィードバックも、よりダイナミックに行える。

6.2 課題

信頼度による解析の制御について

今回は、入力文(ワード)、構文規則、エッジに付与したScore(信頼度)をほとんど同列に扱い、解析の制御を行っている。本来、これらのScoreはまったく性質の異なるものであり、Scoreの算出も複雑な計算を要するはずである。課題としてまず、Scoreの適正な付与方法、利用方法を検討しなければならない。信頼度が信頼できるようになれば、幅切り(:Beam search)による冗長処理の回避や正答率の向上が期待できる。

ポーズユニット対応機能について

試作プログラムでは、結局、文頭部分のCHARTに順次文要素を加えて解析を進める形になる。これでは、信頼度の高い部分から解析を始めるIsland Drivenな解析の効果が薄れてしまう危惧がある。今後は、次の2点を検討する必要があるだろう。

・ポーズユニット毎に解析を終了した後、ユニットを組み立てる処理の有効性
・信頼度の非常に高いユニットが現れた場合の先行ユニットに対するバックトラック処理の実現可能性と効率

音声認識システムとの連携について

基本的に必要となるであろう機能は、今回確認できたが、認識ー解析間のインタフェースについて早急に検討する必要がある。

参考文献

[1]「自然言語処理の基礎技術」野村浩郷著(社団法人　電子情報通信学会)ISBN4-88552-070-3

[2]「Natural Language Processing in PROLOG(An Introduction to Computational Linguistics)」Gerald Gazdar,Chris Mellish著(Addison-Wesley Publishing Company)ISBN0-201-18053-7

[3]"Bidirectional charts:a potential technique for parsing spoken natural language sentences" Oliviero Stock,Computer Speech and Language(1989)3,219-237

付録　関数概要

```
/*****************************************************************************/
/*      FILE-ID          : "chart.c"                                         */
/*      CONTENTS         : 双方向チャートパーザーのプロトタイププログラム    */
/*****************************************************************************/
```

■SHORT main(SHORT argc, PCHAR argv[])

[機能]
チャートパーザーのmain関数。
[オプション]
-g 文法ファイル名(デフォルトは"grammar5.dat")
-s 入力文ファイル名(デフォルトは"sentence.dat")
-p(エッジ等表示時にScoreも表示する)

■SHORT InitializeChartParser(VOID)

[機能]
InitializeChart()、InitializeAgenda()を呼び出す。
[戻り値]
SUCCESS(=0)　　: 成功

■SHORT InitializeChart(VOID)

[機能]
入力文のワード数にあわせてVERTEXテーブルを確保し初期化する。
[戻り値]
SUCCESS(=0)　　: 成功

■SHORT InitializeAgenda(VOID)
[機能]
入力文の各ワードからScoreの高いものをINITIAL_ISLAND_NUM個選ぶ。選んだワードを右辺にもつルール(TERMINAL_SYMBOL)からINACTIVE_EDGEを生成し、AGENDAへ加える。
[戻り値]
SUCCESS(=0)　　: 成功

■SHORT BidirectionalChartParser(VOID)
[機能]
解析処理のメインループ。AGENDAから取り出したエッジXのTypeにより処理を分岐する。
INACTIVE_EDGE:
 CHARTへエッジXを加える。
 CHART上でエッジXと結合できるACTIVE_EDGEを探し、あればそれから新たなエッジを生成しAGENDAへ入れる。
 エッジXのLabelを右辺に含む構文規則を探し、あればその構文規則から新たなエッジを生成しAGENDAへ入れる。(ボトムアップ解析的)
ACTIVE_EDGE:
 CHARTへエッジXを加える。
 CHART上でエッジXと結合できるINACTIVE_EDGEを探し、あればそれから新たなエッジを生成しAGENDAへ入れる。
 エッジXの未処理標識を左辺に構文規則を探し、あればその構文規則から新たなエッジを生成しAGENDAへ入れる。(トップダウン解析的)
[戻り値]
SUCCESS(=0)　　: 成功

■SHORT AICombine(cpPEDGE pEdge)
[機能]
pEdgeの左右双方にある結合可能なエッジを探し、あればINACTIVE_EDGEをACTIVE_EDGEの娘とした形の新しいエッジを生成する。新エッジは、すぐCHARTには加えずにAGENDAへ入れる。なお、pEdgeがTERMINAL_SYMBOLルールからなるACTIVE_EDGEの場合は、結合処理は行わず、ルール右辺のStringが入力文中の適当な位置にあるかをチェックし、あれば当該エッジのTypeをINACTIVE_EDGEに更新してAGENDAへ入れる。
[戻り値]
SUCCESS(=0)　　: 成功

■SHORT NewEdgeFromIR(cpPEDGE pEdge)
[機能]
pEdgeのLabelに等しい標識を右辺のどこかにもつ構文規則から新しいエッジを生成し、AGENDAへ入れる。なお、新エッジのルール右辺のうちpEdgeのLabelに相当する部分は、処理済みとする。
[戻り値]
SUCCESS(=0)　　: 成功

■SHORT NewEdgeFromAR(cpPEDGE pEdge)
[機能]
pEdgeのルール右辺のうち処理済み部分直近の未処理標識に等しい標識を左辺にもつ構文規則から新しいエッジを生成し、AGENDAへ入れる。新エッジは、StartVertex=FinishVertexのACTIVE_EDGEとなる。
[戻り値]
SUCCESS(=0)　　: 成功

■SHORT CreateEdge(cpEDGETYPE Type, SHORT StartVertex, SHORT FinishVertex,
　　cpCATEGORY Label, cpPCATEGORY_L pToFind, cpPCATEGORY_L pFound,
　　PVOID pContents, cpPDAUGHTER_L pDaughter, cpPEDGE *ppEdge)
[機能]
引き数の内容を持つエッジを生成する。
[戻り値]
SUCCESS(=0)　　: 成功

■SHORT CopyEdge(cpPEDGE pEdge, cpPEDGE *ppNewEdge)
[機能]
pEdgeと同じ内容を持つエッジを生成する。
[戻り値]
SUCCESS(=0)　　: 成功

■SHORT DestroyEdge(cpPEDGE pEdge)

[機能]
pEdgeが使用しているメモリエリアを解放する。
[戻り値]
SUCCESS(=0)　　: 成功

■SHORT CheckEdgeExist(cpPEDGE pEdge)
[機能]
CHARTおよびAGENDA上にpEdgeと同じかさらに処理の進んだ状態のエッジが存在するかをチェックする。
チェック内容:
- Labelの一致
- pRuleの一致
- StartVertexの一致もしくはpEdgeのStartVertexの方が大きい
- FinishVertexの一致もしくはpEdgeのFinishVertexの方が小さい
- FromPositionの一致もしくはpEdgeのFromPositionの方が大きい
- ToPositionの一致もしくはpEdgeのToPositionの方が小さい
- pDaughter(娘エッジ)リストの一致もしくはpEdgeの方が部分リストになっている

[戻り値]
TRUE(=1)　　: 真(既存)
FALSE(=0)　　: 偽

■SHORT CheckTerminalSymbolFromEdge(cpPEDGE pEdge)
[機能]
入力文中の適当な位置(pEdgeのStartVertex、FinishVertex相当の位置)に、pEdgeのルール右辺のString(終端記号)が存在するかをチェックする。
[戻り値]
TRUE(=1)　　: 真(既存)
FALSE(=0)　　: 偽

■SHORT AddZDaughterL(cpPDAUGHTER_L *ppDaughter, cpPEDGE pEdge)
[機能]
AllocしたcpDAUGHTER_L構造体にpEdgeを入れ、娘エッジリスト(先頭:*ppDaughterL)の末尾に追加する。*ppDaughterLがNULLの場合、cpDAUGHTER_L構造体のアドレスを*ppDaughterLに返す。
[戻り値]
SUCCESS(=0)　　: 成功

■SHORT AddADaughterL(cpPDAUGHTER_L *ppDaughter, cpPEDGE pEdge)
[機能]
AllocしたcpDAUGHTER_L構造体にpEdgeを入れ、娘エッジリスト(先頭:*ppDaughterL)の先頭に追加する。*ppDaughterLがNULLの場合、cpDAUGHTER_L構造体のアドレスを*ppDaughterLに返す。
[戻り値]
SUCCESS(=0)　　: 成功

■SHORT PutEdgeIntoAgenda(cpPEDGE pEdge)
[機能]
AGENDAにpEdgeを入れる。挿入位置は、pEdgeのScoreをもとに降順にソートし決定する。同Scoreのエッジが存在する場合、pEdgeが次順となる。
[戻り値]
SUCCESS(=0)　　: 成功

■SHORT GetEdgeFromAgenda(cpPEDGE *ppEdge)
[機能]
AGENDAから最もScoreの高いエッジをひとつ取り出す。
[戻り値]
SUCCESS(=0)　　: 成功
FAIL(=!SUCCESS)　: 失敗(AGENDAが空の場合)

■SHORT PutEdgeIntoChart(cpPEDGE pEdge)
[機能]
CHARTの末尾にpEdgeを追加するとともに、VERTEXテーブルのStartVertexに該当するVERTEXにOutGoingEdgeとして登録し、FinishVertexに該当するVERTEXにInComingEdgeとして登録する。
[戻り値]
SUCCESS(=0)　　: 成功

■SHORT ScoreAlCombine(cpPEDGE pAEdge, cpPEDGE pIEdge, PLONG pScore)　　:試行

[機能]
ACTIVE_EDGEとINACTIVE_EDGEの結合により生成するエッジのScoreを計算する。
[戻り値]
SUCCESS(=0)　　: 成功

■SHORT ScoreNewEdgeFromIR(cpPEDGE pIEdge, cpPRULE pRule, PLONG pScore)　　:試行
[機能]
INACTIVE_EDGEと構文規則から生成するエッジのScoreを計算する。
[戻り値]
SUCCESS(=0)　　: 成功

■SHORT ScoreNewEdgeFromAR(cpPEDGE pAEdge, cpPRULE pRule, PLONG pScore)　　:試行
[機能]
ACTIVE_EDGEと構文規則から生成するエッジのScoreを計算する。
[戻り値]
SUCCESS(=0)　　: 成功

```
/*****************************************************************************/
/*      FILE-ID          : "chart_util.c"                                    */
/*      CONTENTS         : 双方向型チャートパーザーのユーティリティプログラム  */
/*****************************************************************************/
```
■SHORT AllocMemory(ULONG AreaSize, PVOID *ppAllocArea)
[機能]
AreaSizeバイトの領域を確保し、そのアドレスを*ppAllocAreaに返す。
[戻り値]
SUCCESS(=0)　　: 成功
FAIL(=!SUCCESS) : 失敗

■SHORT ReallocMemory(ULONG AreaSize, PVOID pOldAllocArea, PVOID *ppNewAllocArea)
[機能]
pOldAllocAreaをAreaSizeバイトの領域として再確保し、そのアドレスを*ppNewAllocArea
に返す。
[戻り値]
SUCCESS(=0)　　: 成功
FAIL(=!SUCCESS) : 失敗

■SHORT FreeMemory(PVOID pFreeArea)
[機能]
pFreeAreaの指す領域を解放する。
[戻り値]
SUCCESS(=0)　　: 成功
FAIL(=!SUCCESS) : 失敗(pFreeAreaがNULLの場合)

■SHORT AddString(PCHAR pString, PCHAR *ppAllocArea)
[機能]
pStringの先頭からヌル文字('¥0')まで分の領域を確保し、文字列をその領域にコピーする。
領域のアドレスを*ppAllocAreaに返す。
[予定]
将来的には文字列の管理を行い、等しい文字列には同じアドレスを返す。
[戻り値]
SUCCESS(=0)　　: 成功
[注意]
領域の確保に失敗した場合、プロセスを終了する。

■SHORT StringCompare(PCHAR pString1, PCHAR pString2)
[機能]
文字列pString1とpString2を比較する。一致した場合、TRUEを返す。
[戻り値]
TRUE(=1)　　: 一致(Match)
FALSE(=0)　　: 不一致(UnMatch)

■SHORT StringCompareNotAa(PCHAR pString1, PCHAR pString2)
[機能]
アルファベット大/小文字の区別をせずに、文字列pString1とpString2を比較する。一致した場合、TRUEを返す。

[戻り値]
TRUE(=1)　　: 一致(Match)
FALSE(=0)　　: 不一致(UnMatch)

```
/*****************************************************************************/
/*      FILE-ID          : "read_datafile.c"                                 */
/*      CONTENTS         : 辞書&文法データの読み込み                         */
/*      SPECIAL THANKS   : Mr.Hayashi                                        */
/*****************************************************************************/
```

■SHORT ReadSentenceFile(PCHAR pFileName, cpPSENTENCE *ppSentence)
[機能]
入力文ファイルを読み込み、cpSENTENCE構造体のリストに展開する。先頭の構造体へのポインタを*ppSentenceに返す。
サブ関数GetWord()がピリオド('.')までの文字列を順次切り出すので、入力文はピリオドまでの文となる。なお、文中の英大文字は小文字に変換する。
[戻り値]
TRUE(=1)　　: 一致(Match)
FALSE(=0)　　: 不一致(UnMatch)
[注意]
単語辞書にない単語が文中に現れた場合、プロセスを終了する。
構造体領域の確保に失敗した場合、プロセスを終了する。

■SHORT ReadGrammarFile(PCHAR pFileName, cpPRULE *ppFirstRule,
　　cpPWORD *ppFirstWord)
[機能]
文法ファイルを読み込み、cpRULEおよびcpWORD構造体のリストに展開する。先頭の構造体へのポインタをそれぞれ*ppFirstRule,*ppFirstWordに返す。
[書式]
句構造規則　　:RULE 標識 -> 標識列;
単語　　:WORD 文字列 = 標識;

標識　　:S | NP | VP | PP | N | IV | TV | P | Det
標識列　　:標識 [標識列]
文字列　　:(任意の単語を表す文字列)
[戻り値]
SUCCESS(=0)　　: 成功

■PCHAR GetWord(PCHAR pBuf, PCHAR pBufEnd, CHAR String[])
[機能]
領域pBufの先頭から区切り文字(改行'¥n',タブ'¥t',ブランク' ',カンマ',')、および文法終了文字(セミコロン';')、文終了文字(ピリオド'.')までを切り出し、Stringに返す。
先頭が区切り文字で始まる場合は、区切り文字以外が現れるまで読み飛ばす。
先頭が文法終了文字で始まる場合は、文法終了文字のみを切り出し、Stringに返す。
先頭が文終了文字で始まる場合は、バッファの終了と同じくNULLを関数戻り値とする。
先頭が'#'で始まる場合は、コメント行と解釈し、改行文字'¥n'までを読み飛ばす。
[戻り値]
次回のpBuf　　: 読み込んだバッファ内容の次バイト目へのポインタ
[注意]
文法終了文字および文終了文字が先頭以外に現れた場合は、バッファポインタを1戻し(=次回の先頭になるようにして)、そこまでの文字列をStringに返す。

■static SHORT　GetCategoryFromString(PCHAR pString, cpPCATEGORY pCategory)
[機能]
標識文字列pStringから内部処理用の標識番号を*pCategoryに返す。
[戻り値]
SUCCESS(=0)　　: 成功
FAIL(=!SUCCESS)　: 失敗(該当する標識文字列がなかった場合)

■static SHORT　InitializeGrammarScore(VOID)　　:試行
[機能]
各構文規則にScoreを付与する。同じ左辺を持つルールのScore合計が100となるよう調整する。
[戻り値]
SUCCESS(=0)　　: 成功

```
/*****************************************************************************/
/*      FILE-ID          : "test_print.c"                                    */
/*      CONTENTS         : チャートパーザー構造体プリント(テスト用)          */
/*****************************************************************************/
■SHORT GetScore(PLONG aScore, USHORT ArayNum)
```
[機能]
乱数を利用し、ArayNum分のScoreを配列(先頭aScore)に格納する。
[戻り値]
SUCCESS(=0)   : 成功

208z
```
/*****************************************************************************/
/*      FILE-ID          : "test_print.c"                                    */
/*      CONTENTS         : チャートパーザー構造体プリント(テスト用)          */
/*****************************************************************************/
■SHORT PrintCHART(cpPSENTENCE pSentence, cpPEDGE pEdge)
```
[機能]
入力文、解析結果、TERMINAL_SYMBOL_EDGE、INACTIVE_EDGE、ACTIVE_EDGEを表示する。
208z
```
■static VOID   DrawEDGE(cpPEDGE pEdge)
```
[機能]
エッジを表示(図)する。
214z
```
■static VOID   DrawDaughter(cpPDAUGHTER pDaughter)
```
[機能]
娘エッジを再帰的に表示(図)する。
-izxrmail
```
■SHORT PrintRULE(cpPRULE pRule)
```
[機能]
文法ルールリストをpRuleから順に表示する。

```
■SHORT PrintSENTENCE(cpPSENTENCE pSentence)
```
[機能]
入力文(cpSENTENCE構造体リスト)をpSentenceから順に表示する。