TR-IT-0332

# ATR-MATRIX 発話分割モジュール説明書

# The document for utterance splitting module bunkatsu in ATR-MATRIX

中嶋 秀治
Hideharu Nakajima

大槻 直子*
Naoko Ohtsuki

林 輝昭*
Teruaki Hayashi

リーブス ベンジャミン*
Benjamin Reaves

竹澤 寿幸
Toshiyuki Takezawa

横尾 昭男
Akio Yokoo

2000.2.8

自然な会話では、1回の発話（発声）の中で、複数の文が一息に話される場合がある。このような発話は翻訳前に文などの言語処理の単位へ分割することが必要となる。ATR-MATRIX では、sprec97 配下の bunkatsu というモジュールで、発話の分割を行なっている。本稿ではその bunkatsu モジュールの概要について説明する。このモジュールは単体または ATR-MATRIX から利用できる。



*ATR 開発室

# ATR-MATRIX 発話分割モジュール説明書

1999-01-06 初版
1999-03-31 改訂
2000-02-07 改訂

## 1 機能概要

翻訳や解釈のための自然言語処理単位を「文」と呼ぶことにすると、話者は、自身の発話権を保持している間に複数の「文」を話す場合がある。

これまで、複数の「文」からなる発話の音声認識結果は単語の系列のままであり、各「文」の境界は認識されていなかった。そのため、このような音声認識結果や「文」としての境界が明示されていない入力文に対しては、翻訳を行なう前に、それらを「文」へ分割する必要があった。

このモジュールは、"bunkatsu" と呼ばれ、音声認識結果の単語間に「文」の境界が入り得るか否かを判定し、判定の結果、入り得る場合には、単語の系列に境界の記号を挿入する。すなわち、2 つ以上の「文」からなる発話を「文」ごとに分割する。

現在のところ、認識結果データの WORDS= の中で、文と文との境界には SENT-END/SENT-START が、UTT-START の後ろには SENT-START が、UTT-END の前には SENT-END がそれぞれ挿入される。wordids= の中では、SENT-START と SENT-END の位置に、"1" と"2" とがそれぞれ挿入される。また vars= には"1" が、divs= には、"/" が挿入される。times= には、前の文末の単語末までの時間と同じ数値が挿入される。(例は、松田 他著の ATR-ITL テクニカルレポート 「ATR-MATRIX データフォーマット仕様書」の音声認識結果データの項を参照。)

sprec97 内での、このモジュール bunkatsu と音声認識モジュール SPREC との関係は付図の通りである。

## 2 使い方とそれに応じたインストール方法

### 2.1 単体での利用(スタンドアロンでの利用)

#### 2.1.1 コンパイル

make -DSTANDALONE punc
または
make standalone

ただし、コンパイル前に、必要なファイルを punc.c 内で指定する。
日本語の場合、以下の定義を行なう。(英語用については、INPUTE1 等を定義する。)

```
#define INPUTJ1 "./PARAM/JE/heu6.grm"              /* 分割を検証するための規則 */
#define INPUTJ2 "./PARAM/JE/freq-2words.prm"        /* 分割に関する頻度情報 */
#define INPUTJ3 "/DB/SHARE/MASTER_LEX/TDMT/MASTER\_LEXICON" /* マスター辞書 */
#define INPUTJ4 "./SAVE/JE/RECOG_LEX.vn1000"                /* 音声認識辞書 */
#define INPUTJ5 "./PARAM/JE/config.prm"                  /* パラメータファイル */
```

詳しくは、末尾の解説書「発話単位の分割による言語処理単位への変換 1999-02-26 納品」の 4 ページを参照。

#### 2.1.2 使い方

% cat lattice_file | punc (jap|eng)

モードの指定 jap:Japanese eng:English

### 2.2 ATR-MATRIX での利用

ATR-MATRIX の認識サブシステム ATRsprec のモジュール sprec97 の中で、分割処理が実行される。モジュール間の上下関係は次の通り:

トップはSipMainController(ATR-Matrix実行形式ファイル)で、これがSipSprecControllerをforkし、また、SPREC(認識サブシステム)を実行する。SPRECのATRexec（実行形式ファイル）は、複数のモジュールからなり、その1つのモジュールがSprec97で、このモジュールのうちの1つは、音声認識の後処理をおこなう。その後処理をおこなうモジュールの1つとして、（このマニュアルで説明する）分割処理がある。これはサブルーチンで呼ばれる。

このサブルーチン名はsprec97_bunkatsuで、これはpuncのルーチン（このマニュアルで説明する発話分割モジュール）を呼ぶ。

### 2.2.1 コンパイル

巻末の"How to install Sprec97 in ATR-Matrix" のページを参照。

### 2.2.2 使い方

巻末の「sprec97安定版」のページを参照。

実行前に、configファイルの設定が必要である。日本語の場合は以下を指定する。
（例：/home/atm02/matrix/cstar_set/run/config.j）

```
sprec97:j-frequency=$SIP_SPREC97/src/bunkatsu/PARAM/JE/freq-2words.prm
sprec97:j-lexicon=$SIP_RESOURCE/tdmt_j_model/MASTER_LEXICON
sprec97:j-heuristic=$SIP_SPREC97/src/bunkatsu/PARAM/JE/heu6.grm
sprec97:j-recognizerLexicon=$SIP_RESOURCE/sprec_j_model/lmodel/19990205/LEX.W.comp
sprec97:j-environment=$SIP_SPREC97/src/bunkatsu/PARAM/JE/config.prm
```

のように5つの引数に対して、それぞれその右側に書かれているようなファイル名を指定する。
（英語の場合は、sprec97:e-* を指定する。）

上記の設定のあと、ATR-MATRIXを起動する。

## 3 モジュールから参照されるファイルのフォーマット

モジュールから参照されるファイルの指定は前章の方法で行なう。ファイルは全部で5つ存在する。

- 頻度ファイル（ sprec97:j-frequency で指定。 2.1 で説明）
- マスター辞書（ sprec97:j-lexicon で指定。 2.2 で説明）
- 検査用規則 （ sprec97:j-heuristic で指定。 2.3 で説明）
- 音声認識辞書（ sprec97:j-recognizerLexicon で指定。 2.4 で説明）
- 環境ファイル（ sprec97:j-environment で指定。 2.5 で説明）

ここでは、これらのファイルの中身の書き方に関して、

$SAMPLE=$TOP/KUTEN/CLAUSE-OPEN1998/

として順に説明する。（$TOP は関連するファイルを格納したトップのディレクトリ名をさす。CD-ROM の ITL_KUTEN.tgz を展開した場合は、展開したディレクトリ名をさす。）

### 3.1 頻度ファイル

連接する、2つの ” 品詞名 | 活用型 | 活用形” の組の出現頻度の並び。書式の例は

$SAMPLE/PARAM/(JE|EJ)/freq-2words.prm

を参照。以下はその抜粋。

```
<< 2語単位での出現数集計 >>
   比較条件：品詞 & 活用形

分母
C0([ 助動詞 | 五段カ | 基本    ・ 普通名詞 ||              ]) =    11
C0([ 本動詞 | 一段 | 連用      ・ 補助動詞 | 五段カ | 連用 ]) =     8
C0([ 普通名詞 ||              ・ 係助詞 ||               ]) =  3063
        :
C0([ 終助詞 ||                ・ END-OF-TURN             ]) =  4332
        :
分子
C1([ 本動詞 | サ変 | う        ・ 助動詞 | 無変化 | 基本   ]) =     1
C1([ サ変形容名詞 ||          ・ 補助動詞 | 特殊ラ | 命令 ]) =    11
C1([ END-OF-TURN              ・ 本動詞 | 特殊ラ | 基本   ]) =     2
        :
C1([ 格助詞 ||                ・ 感動詞 ||               ]) =    59
C2([ 本動詞 | 特殊サ | 基本    ・ 形容詞 | 形容詞 | 基本   ]) =     3
        :
C2([ 人称接尾辞 ||            ・ サ変名詞 ||             ]) =     3
```

### 3.2 マスター辞書

通常のマスター辞書である。

単語の定義情報（表現形、発音、品詞名、などからなる7つ組）を取得するために用いられる。

### 3.3 検査用規則ファイル

日本語モードのみで使用される。

連接する、4つの” 表層表現 | 品詞名 | 活用形 | 活用型” のならび（4単語のならびに相当）の真ん中に「。」が入るか否かを、”0”（入らない）,”1”（入る）で示したルールの並び。書式の例は

$SAMPLE/PARAM/JE/heu4.grm

を参照。以下はその抜粋。

```
( *|*|*|*     + +|+|*|*          ・ +| 語尾 |+| 連用 + *|*|*|* ){ 0 }
( *| 語尾 |+|+ + +| 語尾 |+| 連用 ・ +|+|*|*          + *|*|*|* ){ 1 }
( *|*|*|* + +| 感動詞 |*|* ・ +| 接続助詞 ||          + *|*|*|* ){ 0 }
( *|*|*|* + +| 感動詞 |*|* ・ +| 終助詞 ||           + *|*|*|* ){ 0 }
( *|*|*|* + +| 感動詞 |*|* ・ +| 形容詞 | 形容詞 | 語幹 + *|*|*|* ){ 0 }
   :
```

記述方法は次の通り。

```
規則        : ( 第一語 + 第二語 ・ 第三語 + 第四語 ){ (0|1) }
            :   先に出現した規則が優先される。
            :   各語と ( + ・ ){ } とのスペースは１つ以上空けること。

各語の規則 ( |||で区切られた部分。 表層表現 | 品詞 | 活用形 | 活用型 )
  (null)    : 0個                  ex. |||
    *       : 0個以上存在する     ex. *|*|*|*
    +       : 1個以上存在する     ex. +|+|*|*
  (xxxx)    : 個別指定             ex. +| 感動詞 ||
  !(xxxx)   : 個別指定の否定       ex. +|! 接続詞 ||

注釈・コメント
  行末の ‘‘} ‘‘ 以降
  行頭が ‘‘( ‘‘ 以外
```

詳細は、以下の文書（末尾に置場所を記載した）を参照のこと

「統計的な言語モデルによる節境界検出ツールの作成」 P.8 ～

## 3.4 音声認識辞書

通常の音声認識用辞書である。

マスター辞書内の単語のすべてが発話に現れるわけではない。必要な単語は、音声認識結果に含まれるものだけであるので、音声認識用の辞書を用いて、単語を絞り込む。

音声認識辞書にも定義情報が書かれている場合があるが、それは単なる注釈用なので内容が正しいかどうかの保証がない。そのため、マスター辞書が必要となる。

## 3.5 環境ファイル

以下の条件を指定する。

```
# 単語条件：表層、品詞、活用型、活用形のどの情報を利用するかを指定する
  KIND      = 2
# 単語組数：発話境界の前後何単語の情報を使うか
  WORD      = 4
# 閾値：境界を置くか否かの判定のための閾値
  SHIKII    = 0.28
# 検査用規則を使うか否か
  HEU       = heu/nor
```

詳細は「発話単位の分割による言語処理単位への変換 1999-02-26 」の P.6 を参照。

## 4　上記ファイルを作成するためのツールの起動法と学習データのフォーマット

対象タスクを変えた場合に、上記の２章で説明したファイルを作成するために、起動が必要となる。
以下の書類の記載内容に従って作成する。

「統計的な言語モデルによる節境界検出ツールの作成 1997-08-29 」

この中の”２．予備実験 (P2 ～ P9)” を参照して次のパラメータを決定する。

- 単語属性
- 単語組数
- 閾値
- 検査用規則 (ヒューリスティック)

ただし検査用規則ファイルについては、修正が必要である。分割位置として認められないものには、ルールの右端を { 0 } に、認められるものには { 1 } に修正する。

# 5 分割プログラム自体の構成

## 5.1 主な流れ

主要部分は punc.c である。ソースにコメントが豊富なので、粗筋のみ書く。

まず、lattice を読み込み、関数 Judge で、単語の間に「。」（分割境界）が入る可能性を示すスコアを計算し、そのスコアが予め定めておいた閾値以上の場合に「。」が入ると仮判定する（判定の手法については、巻末の竹澤ら著の文献を参照）。つぎに、関数 Verify で、その仮判定結果をルールで検証し、ルールに合えば、「。」の箇所と決定する（ただし、この検証は日本語の場合のみ。）

## 6 参考文献

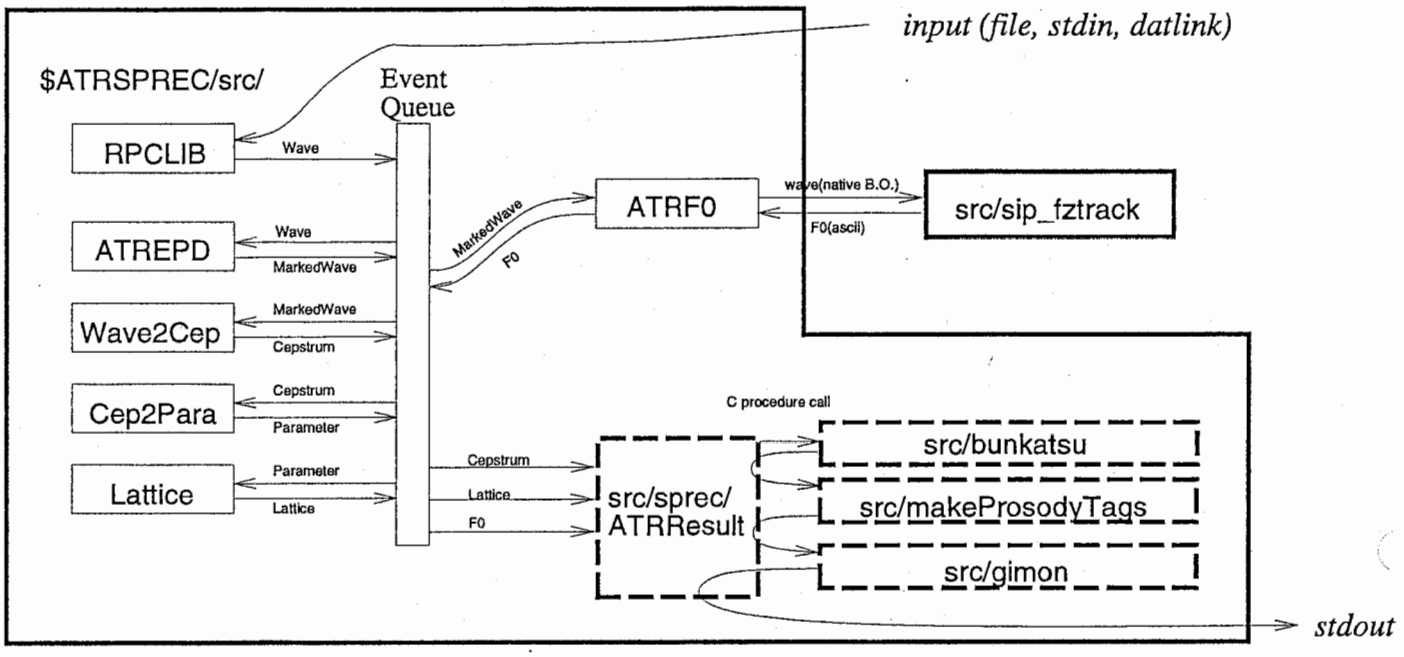
- 研究会資料、論文等
  - 竹澤、森元、”発話単位の分割または接合による言語処理単位への変換”、
    情報処理学会研究会（SLP）１８—４、１９９７。
  - 竹澤、森元、”発話単位の分割または接合による言語処理単位への変換手法”、
    自然言語処理、Vol.6, No.2、１９９９。

- 3章及び4章の tool(csh/awk script etc.) に関する解説
  - 統計的な言語モデルによる節境界検出ツールの作成 1997-08-29 納品
    $DocTop/PRODUCT/1997.08/CLAUSE/*.*
  - 発話単位から言語処理単位への変換プログラム 1998-02-27 納品
    $DocTop/PRODUCT/1998.02/KUTEN/clause.tex
  - 発話単位の分割または接合による言語処理単位への変換 1998-08-31 納品
    $DocTop/PRODUCT/1998.08/KUTEN/clause.tex
  - 発話単位の分割による言語処理単位への変換 1999-02-26 納品
    $DocTop/PRODUCT/1999.02/KUTEN/clause.tex

    これらの文書の中に示されているディレクトリには、今は使われていないものもありますが、すべて、CD-ROM の ITL_PRODUCT.tgz を展開したディレクトリ名に読み替えて下さい。対象になる名称は、次のとおりです。

    ```
    /home/as54
    /home/as403
    /dept4/work4
    など
    ```

# 韻律処理付きSPREC：sprec97




One Unix Process

Lattice — ATRSPREC module, source in $ATRSPREC/src/ATRLATTICE/

src/bunkatsu — sprec97 module, source in sprec97/src/bunkatsu/

See readme.html for installation information

```
--------------------------------------------------------------------------

                    How to Install Sprec97 in ATR-Matrix

--------------------------------------------------------------------------

This is an example of how to install the demo version of Sprec97.
To install and use the Workbench Version, please see workbench.txt


# Set your environment variables to the target disk space

setenv ATRSPREC     ~matrix/resources/SPREC                 # 適当に
setenv LD_LIBRARY_PATH /usr/local/lang/tcl-8.0/lib:/usr/local/lang/tk-8.0/lib
setenv LD_LIBRARY_PATH ${LD_LIBRARY_PATH}:$ATRSPREC/shlib
setenv SIP_SPREC97 ~matrix/resources/sprec97                # 適当に
setenv SIP_TDMT     /DB/SHARE/MASTER_LEX/TDMT               # 発話分割のため


# Get Sprec97 and Sprec if you need it (you need r06r02 or later)

# I assume the above env settings, but their relationship is not necessary
cd ~matrix/resources
cvs -d /home/tsgfs00/sip/repository/sip.cvsroot checkout sprec97
cvs -d ~singer/CVSROOT checkout SPREC
rm -rf $ATRSPREC/src/SSS    # disk spaceとコンパイル時間を節約


# Make the Executables

# (don't use Digital Unix 4.0)

cd $ATRSPREC/src ; make clobber all install
cd $SIP_SPREC97/src ; make NOSPREC=1

# debug (gdbで使える) 版をメイクするには、
# cd $ATRSPREC/src ; make ATRDEBUG=1 clobber all install
# cd $SIP_SPREC97/src ; make NOSPREC=1 debug
# cd $SIP_SPREC97/run.myname ; cp ../test/{*,.*} .
# gdb ./ATRexec

# Verify Installation

mkdir $SIP_SPREC97/run.myname    # (例えば)
cd    $SIP_SPREC97/run.myname
cp    $SIP_SPREC97/test/* .

# Japanese test
../bin/ATRexec -config=config.j > stdout.j.test.txt
diff stdout.j.test.txt ../test/stdout.j.ref.txt

# English test
../bin/ATRexec -config=config.e > stdout.e.test.txt
diff stdout.e.test.txt ../test/stdout.e.ref.txt

# CPU時間とホスト名しか違いがないはず。
# 尚、現在のconfigファイルでは、モデルファイルは絶対pathを使っていますので、
```

```
# ITLのNFSで見えなければ、テストできない。
# 「Can not open IndicatorTk.py」が出たら、unset DISPLAYして、再実行してください。

# Verify AD and Remote Command operation

# 生データとRPC命令で動作確認。
# command.pyはsprec_sendでATRexecを制御します。
# ../etc/commands.py を実行する前に、-rと-hの引数を確認して下さい。
# edit datlink.sprec97   :    適当なコマンドを入れてください。

setenv AUDIO_DEVICE pau03   # DATLINKのある所。datlink.sprec97が使う
setenv DISPLAY pau03:0.0    # commands.py、SPRECのdisplaypowが使う

../etc/commands.py &

../bin/ATRexec '-I/Ocontrol:inputFd=process(datlink.sprec97) \
  -config=config.j \
  -I/Ocontrol:rpcNumber=5 \
  -I/Ocontrol:inputFormat=NoHeader \
  -sprec97:track=0 \
  &

------------------------------------------------------------------------------
For Release Notes, please see releasenotes.txt
For a list of cautions and Bugs, please see buglist.txt
For pretty documentation, cd doc ; make
------------------------------------------------------------------------------
$Header: /home/tsgfs00/sip/repository/sip.cvsroot/sprec97/doc/install.txt,v 1.17
```

# sprec97安定版

```
set r = ben-r06r03
cvs -d /home/tsgfs00/sip/repository/sip.cvsroot checkout -r $r sprec97
```

で出て来ます。

## 特徴 Features

- CTRL recording のサポート
- queueの問題はなくなった。pitch=1の場合のpitch=0の場合も。
- sprec97:track=0すると、実行が速くてメモリに軽い。
- ATRuserへmic-off送ったら喋りながらのアボート機能は効く。
- LinuxもOSF1もHP-UX上で安定にピッチ抽出できます。
- 日本語でも英語でも文分割できます。
- Sprec Satellite Controllerに変更不要

## 注意点 Notes:

- sprec97:track=1はデバッグモードですが、実行が遅くて、サブシェルによってメモリ不足になった。
- 以前のMatrixで使われているSprecバージョン(r06a02)とリンクすると、キャンセル機能は前のまま。又、-lATRcompressをsrc/makefileから消さないとリンクできない。
- リモート・コマンドbunkatsu-on, bunkatsu-offを使っても効かない
- オプションが変りました。

```
# 削除
sprec97:heuristic  # e-heuristic, j-heuristicに分れている
sprec97:frequency
sprec97:lexicon
# 追加
sprec97:merge_file      # SprecのATRresultモジュールのドキュメントに参照
sprec97:e-heuristic     # 英語版heuristicファイル。
sprec97:e-frequency     # (詳しくはsrc/bunkatsu/punc.c参照下さい)
sprec97:e-lexicon
sprec97:e-recognizerLexicon  # ATRlattice:lexiconに一致するべき
sprec97:e-environment
sprec97:j-heuristic     # 同上、日本語の場合
sprec97:j-frequency
sprec97:j-lexicon
sprec97:j-recognizerLexicon
sprec97:j-environment
sprec97:startWithBunkatsuOn=1  # turn on/off 分割処理
                               # (ピッチ処理on/offと独立に)
# 以上のパラメーターの設定の例には、$SIP_SPREC97/test/config.jご参照下さい。
```

- startWithBunkatsuOnは、英語の場合変に分割される可能性がある。
- 英語の場合、ピッチ抽出はできますが、勝手に「？」を入れないので、翻訳結果に

影響はない。prosodiesフィールドしか反映されない。現在のTDMTはそのprosodiesフィールドを処理しない。具体的には、sprec97:language=eの時、sprec97:startWithPitchOn=0 にしてください。

- startWithBunkatsuOnは、英語の場合変に分割される可能性がある。

インストールとテストは、 sprec97/doc/install.txt に参照下さい。
dingpitchと以前のsip_fztrackの説明は、 sprec97/doc/dingpitch.texから html 又は ps
Matrix全体のTechnical Report の下書きは、 sprec97/doc/matrixTR.texから html 又は ps
Sprec97の設計の説明はMicrosfot-Word-98フォーマットで ここ にあります。

---

Ben Reaves ..... tel 1315 ..... 2nd floor, northwest corner ..... ben@itl
$Header: /home/tsgfs00/sip/repository/sip.cvsroot/sprec97/doc/ad.html,v 1.24 199

目　次

（フルパス指定の無いファイルは/home/as54/nohtsuki/KUTEN/CLAUSE 以下）

音声認識実験で使用している音声言語データベース（ＳＬＤＢ）は、処理単位が発話単位（通訳者に渡す区切り）なので、複数の文が含まれていることがある。
部分木出力ＳＳＳＬＲ音声認識装置では処理することができるが、後工程の翻訳システム（ＴＤＭＴ）では、文を処理単位とするため、複数の文は翻訳できない。
音声認識結果には現在、句点「。」が付加されていないので、句点の位置を示す必要がある。そこで、英文について研究された、統計的な言語モデルによる節境界予測の手法が、和文でも有効かどうかを調査した。

## １．節境界検出手法

文献[1]の手法を説明する。

### ａ．訓練セットの単語から、必要な統計量を算出しておく。

文頭から単語間を１つずつ走査し、前後２単語毎の統計量を求める。

(1) F([w1 w2・]): バイグラム [w1 w2] の右に節境界が現れる頻度

C([wi wj・])をバイグラム[wi wj]の右に節境界が現れる回数とし、
C([wi wj]) をバイグラム[wi wj]が訓練セットに現れる総数とすれば、
F([wi wj・])は次の式となる。

$$F([wi\ wj\cdot]) = \frac{C([wi\ wj\cdot])}{C([wi\ wj])}$$

(2) F([w2・w3 ]): バイグラム [w2 w3] の中間に節境界が現れる頻度
(3) F([・w3 w4]): バイグラム [w3 w4] の左に節境界が現れる頻度

(2)(3)とも(1)と同様の方法で求める。

### ｂ．テストセットの単語列に訓練セットの統計量を適用し、推定頻度を求める。

テストセットの４単語組みに対して、発話を分割するのにもっともらしい個所かどうか決定するために、文頭から単語間を１つずつ走査し、その前後２単語（４単語）を参照していく。単語列を[w1 w2・w3 w4] とした場合、w2とw3 の間に節境界がある確率を求めるには、次の式のように２単語毎の統計量を利用して推定頻度 F~([w1 w2・w3 w4])を求める。

$$F\sim([w1\ w2\cdot w3\ w4]) = \frac{C([w1\ w2\cdot]) + C([w2\cdot w3]) + C\{[\cdot w3\ w4])}{C([w1\ w2]) + C([w2\ w3]) + C\{[w3\ w4])}$$

### ｃ．推定頻度を評価して節境界を示す。

F~ の値が閾値より大きければそこを節境界とする。
結果は以下のように分類する。

(1) 節境界である場所で規則が成功する － 当たり(正解) [○]
(2) 節境界である場所で規則が失敗する － はずれ(誤り) [×]
(3) 節境界でない場所で規則が失敗する － 却下成功(正解) [↓]
(4) 節境界でない場所で規則が成功する － 涌き出し誤り(警告) [※]

閾値の値は、小さくすると当たり[○]が増えるが涌き出し誤り[※]も増える。大きくすると涌き出し誤り[※]は減るが、はずれ[×]が増える。全体として正解率が最大になるように、あらかじめ人手で調整しておく。

## 2．予備実験

### a．使用した単語列

音声言語データベース（ＳＬＤＢ）のＪＭＯＲを使用した。
テストセットは、専用のディレクトリ(ここでは JMOR.9.test/)に９会話のＪＭＯＲファイルを格納しておく。
訓練セットは、専用のディレクトリ(JMOR.9.learn/)にＳＬＤＢから９会話を除いたＪＭＯＲファイルを格納しておく。
実行時は、次のように設定する。

---

（例） freq-9.csh

```
set JMORLEARN = JMOR.9.learn      # 訓練セット   Directory
                ~~~~~~~~~~~~
set JMORTEST  = JMOR.9.test       # テストセット Directory
                ~~~~~~~~~~~~
set CHECK     = $JMORTEST/JMOR.lst # テストセット リスト
                |
set LOG       = freq-9
ls $JMORTEST/*.JMOR >$CHECK
#
foreach KIND (0 1 2)      # 0:品詞 1:品詞＆活用 2:表層＆品詞＆活用
                |
    freq.awk -f ./get_key.awk -v Kind=$KIND -v Wordnum=$WORD
      -v Check=$CHECK $JMORLEARN/*.JMOR | freq-est.awk -f ~/awk/rev_index.awk
               ~~~~~~ ~~~~~~~~~~~~~~~~~
      -f split_4words.awk -f heuristic.awk -v Threshold=$SHIKII -v Dir=$DIR
      -v Heuristic=$HEUFILE >$DIR/$LOG.$KIND.est
                |
end
```

---

※ freq-9.csh 内で使用していたツールの仕様は次のとおり。

```
freq.awk -f ./get_key.awk
         -v Kind=    単語属性      0:品詞 1:品詞＆活用 2:表層＆品詞＆活用
         -v Wordnum=単語組み数     3 or 4
         -v Check=  テストセットディレクトリ名
         入力                      訓練セット名(複数可)
         > 出力                    結果ファイル名

freq-est.awk -f ~/awk/rev_index.awk -f split_4words.awk -f heuristic.awk
         -v Threshold=閾値
         -v Dir=      データ出力ディレクトリ名
         -v Heuristic=ヒューリスティック規則の記述ファイル名
         入力
         > 出力                    評価結果ファイル名
```

b．単語属性

ＪＭＯＲでは、１単語毎に 表層|読み|品詞|活用形|活用型|音便 という属性が設定されている。単語の条件として妥当なものを調べる実験を行なった。実行時は、次のように設定する。

```
(例)  freq-9.csh
          |
foreach KIND (0 1 2)       # 0:品詞 1:品詞＆活用 2:表層＆品詞＆活用
              ~~~~~
          |
    freq.awk -f ./get_key.awk -v Kind=$KIND -v Wordnum=$WORD
                                        ~~~~~
      -v Check=$CHECK $JMORLEARN/*.JMOR | freq-est.awk -f ~/awk/rev_index.awk
      -f split_4words.awk -f heuristic.awk -v Threshold=$SHIKII -v Dir=$DIR
      -v Heuristic=$HEUFILE >$DIR/$LOG.$KIND.est

          |
end
```

(1)品詞　(freq-9.0.thr0.10.w4.nor/freq-9.0.est)
--<大計>-- 会話数: 166 句点: 289 節境界: 123
○:276 ×:13 ↓:2320 ※:345 再現率:0.955017 適合率:0.444444
正解数:2596 全語数:2954 合計正解率:0.878808 平均正解率:0.699731
~~~~~~~~ ~~~~~~~~

(2)品詞＆(活用形・活用型)　(freq-9.1.thr0.10.w4.nor/freq-9.1.est)
--<大計>-- 会話数: 166 句点: 289 節境界: 123
○:285 ×:4 ↓:2417 ※:248 再現率:0.986159 適合率:0.534709
正解数:2702 全語数:2954 合計正解率:0.914692 平均正解率:0.760434
~~~~~~~~ ~~~~~~~~

(3)表層＆品詞＆(活用形・活用型) (freq-9.2.thr0.10.w4.nor/freq-9.2.est)
--<大計>-- 会話数: 166 句点: 289 節境界: 123
○:281 ×:8 ↓:2411 ※:254 再現率:0.972318 適合率:0.525234
正解数:2692 全語数:2954 合計正解率:0.911307 平均正解率:0.748776
~~~~~~~~ ~~~~~~~~

以上から、(2)品詞 ＆ (活用形・活用型) の正解率が高いことがわかった。
~~~~~~~~

c．単語組数

文献[1]の手法は4単語組みであるが、結果を個別に検討したところ、最初の3単語で判定が決まってしまうようであった。そこで3単語組みでも実験を試みた。推定頻度は、次の式で求める。

$$F\sim([w1\ w2\cdot w3\ ]) = \frac{C([w1\ w2\cdot\ ]) + C([w2\cdot w3])}{C([w1\ w2]) + C([w2\ w3])}$$

実行時は、次のように設定する。

```
(例)  freq-9.csh
                          |
set WORD       = 3                    # 単語組数
                 ~~~
                          |
foreach KIND (1)       # 0:品詞 1:品詞＆活用 2:表層＆品詞＆活用
    set DIR = $LOG.$KIND.thr$SHIKII.w$WORD.$HEU
    if(! -e $DIR) mkdir $DIR
    freq.awk -f ./get_key.awk -v Kind=$KIND -v Wordnum=$WORD
                                                      ~~~~~
      -v Check=$CHECK $JMORLEARN/*.JMOR | freq-est.awk -f ~/awk/rev_index.awk
      -f split_4words.awk -f heuristic.awk -v Threshold=$SHIKII -v Dir=$DIR
      -v Heuristic=$HEUFILE >$DIR/$LOG.$KIND.est
                          |
end
```

(1) 3単語組み (freq-9.1.thr0.10.w3.nor/freq-9.1.est)

```
  --<大計>-- 会話数:  166  句点:  289  節境界:  123
     ○:285 ×:4 ↓:2486 ※:179  再現率:0.986159 適合率:0.614224
     正解数:2771  全語数:2954   合計正解率:0.938050 平均正解率:0.800192
                                          ~~~~~~~~          ~~~~~~~~
```

(2) 4単語組み (freq-9.1.thr0.10.w4.nor/freq-9.1.est)

```
  --<大計>-- 会話数:  166  句点:  289  節境界:  123
     ○:285 ×:4 ↓:2417 ※:248  再現率:0.986159 適合率:0.534709
     正解数:2702  全語数:2954   合計正解率:0.914692 平均正解率:0.760434
                                          ~~~~~~~~          ~~~~~~~~
```

以上から、(1) 3単語組み の正解率が高いことがわかった。

d．閾値

閾値を推移させながら、正解率が最高値になる地点を模索した。
方法は<<閾値の決定について>>参照　　<threshold.doc>

<< 閾値による正解率の変化 >>

| 閾値 | ○ | × | ↓ | ※ | 再現率 ○ / (○+×) | 適合率 ○ / (○+※) | 平均正解率 (再現+適合) / 2 | 合計正解率 (○+↓) / (○+×+↓+※) |
|---|---|---|---|---|---|---|---|---|
| 0.10 | 285 | 4 | 2486 | 179 | 0.986159 | 0.614224 | 0.800192 | 0.938050 |
| 0.20 | 278 | 11 | 2516 | 149 | 0.961938 | 0.651054 | 0.806496 | 0.945836 |
| 0.30 | 275 | 14 | 2520 | 145 | 0.951557 | 0.654762 | 0.803159 | 0.946175 |
| 0.31 | 275 | 14 | 2520 | 145 | 0.951557 | 0.654762 | 0.803159 | 0.946175 |
| 0.32 | 275 | 14 | 2520 | 145 | 0.951557 | 0.654762 | 0.803159 | 0.946175 |
| 0.33 | 275 | 14 | 2520 | 145 | 0.951557 | 0.654762 | 0.803159 | 0.946175 |
| 0.34 | 275 | 14 | 2567 | 98 | 0.951557 | 0.737265 | 0.844411 | 0.962085 |
| 0.35 | 275 | 14 | 2567 | 98 | 0.951557 | 0.737265 | 0.844411 | 0.962085 |
| 0.36 | 275 | 14 | 2567 | 98 | 0.951557 | 0.737265 | 0.844411 | 0.962085 |
| 0.37 | 275 | 14 | 2598 | 67 | 0.951557 | 0.804094 | 0.877825 | 0.972580 |
| 0.38 | 274 | 15 | 2599 | 66 | 0.948097 | 0.805882 | 0.876990 | 0.972580 |
| 0.39 | 274 | 15 | 2599 | 66 | 0.948097 | 0.805882 | 0.876990 | 0.972580 |
| 0.40 | 274 | 15 | 2599 | 66 | 0.948097 | 0.805882 | 0.876990 | 0.972580 |
| 0.41 | 274 | 15 | 2603 | 62 | 0.948097 | 0.815476 | 0.881787 | 0.973934 |
| 0.42 | 273 | 16 | 2606 | 59 | 0.944637 | 0.822289 | 0.883463 | 0.974611 |
| 0.43 | 273 | 16 | 2608 | 57 | 0.944637 | 0.827273 | * 0.885955 | * 0.975288 |
| 0.44 | 273 | 16 | 2608 | 57 | 0.944637 | 0.827273 | * 0.885955 | * 0.975288 |
| 0.45 | 251 | 38 | 2615 | 50 | 0.868512 | 0.833887 | 0.851200 | 0.970210 |
| 0.46 | 170 | 119 | 2615 | 50 | 0.588235 | 0.772727 | 0.680481 | 0.942789 |
| 0.47 | 159 | 130 | 2615 | 50 | 0.550173 | 0.760766 | 0.655469 | 0.939066 |
| 0.48 | 145 | 144 | 2615 | 50 | 0.501730 | 0.743590 | 0.622660 | 0.934326 |
| 0.49 | 145 | 144 | 2615 | 50 | 0.501730 | 0.743590 | 0.622660 | 0.934326 |
| 0.50 | 143 | 146 | 2624 | 41 | 0.494810 | 0.777174 | 0.635992 | 0.936696 |
| 0.60 | 125 | 164 | 2635 | 30 | 0.432526 | 0.806452 | 0.619489 | 0.934326 |
| 0.70 | 118 | 171 | 2652 | 13 | 0.408304 | 0.900763 | 0.654534 | 0.937712 |
| 0.80 | 116 | 173 | 2654 | 11 | 0.401384 | 0.913386 | 0.657385 | 0.937712 |
| 0.90 | 44 | 245 | 2664 | 1 | 0.152249 | 0.977778 | 0.565013 | 0.916723 |

＊：正解率の最高値。

平均正解率、合計正解率とも閾値を 0.43 または 0.44 にすれば、最高値を得られるということになる。
この場合、0.42 と 0.45 の正解率を参照すると、0.42 の方が近い正解率なので、0.43 が採用された。

e．ヒューリスティックの導入

品詞＆活用を単語の条件としたことにより、同じ品詞で異なる単語による副作用が起こる場合がある。
例えば、「申し訳ございません|感動詞」の場合、単独で使われる時と、「申し訳ございません|感動詞 が|接続助詞」のように続きがある時とがある。感動詞の場合、他の単語が「はい」「ええ」「あ」のように単独でしか使われないものが圧倒的に多いため、続きがある時でも節境界として判断されてしまう。そこで、ヒューリスティックを導入し、このようなケースを救済することにする。

ヒューリスティックの規則は、専用のファイル(heuristic.grm)にテキスト形式で記述する。記述方法は次のように定める。

```
<< ヒューリスティックの規則 >>                                    1997-07-15.Tue
##############################################################################
# 規則          : ( 第一語 + 第二語 ・ 第三語 + 第四語 ){ 旧記号→新記号 }
#               :   先に出現した規則が優先される。
#
# 各語の規則( |||で区切られた部分。 表層表現|品詞|活用形|活用型 )
#   (null)      : 0個                ex. |||
#     *         : 0個以上存在する    ex. *|*|*|*
#     +         : 1個以上存在する    ex. +|+|*|*
#   (xxxx)      : 個別指定           ex. +|感動詞||
#   !(xxxx)     : 個別指定の否定     ex. +|!接続詞||
#
# 注釈・コメント
#   行末の "} " 以降
#   行頭が "( " 以外
#
# 注意点        :   各語と ( + ・ ){ } とのスペースは1つ以上空けること。(必須)
#               :   旧記号→新記号が反例の場合はスペース2つ空けて記述する。
##############################################################################

< 規則の例 >

  0．共通
( *|*|*|* + +|+|*|* ・ ||| + ||| ){ ×→○ ↓→○ }
==============================================================================

  2．感動詞
( *|*|*|* + +|感動詞|| ・ +|接続助詞|| + *|*|*|* ){ ※→↓ } ;申し訳ございません・
が
( *|*|*|* + +|感動詞|| ・ +|+|*|*      + *|*|*|* ){ ※→○ ×→○ }
==============================================================================
```

実行時は次のように指定する。

```
(例)  freq-9.csh
                          |
set HEU        = heu                # ヒューリスティック
                 ~~~                #   nor(mal):なし heu(ristic):あり
set HEUFILE    = heuristic.grm      # ヒューリスティック 規則
                 ~~~~~~~~~~~~~~
                          |
foreach KIND (1)
    set DIR = $LOG.$KIND.thr$SHIKII.w$WORD.$HEU
                                                ~~~~
    if(! -e $DIR) mkdir $DIR
    freq.awk -f ./get_key.awk -v Kind=$KIND -v Wordnum=$WORD -v Check=$CHECK $JMORLE
ARN/*.JMOR | freq-est.awk -f ~/awk/rev_index.awk -f split_4words.awk -f heuristic.aw
k -v Threshold=$SHIKII -v Dir=$DIR -v Heuristic=$HEUFILE
                                                              ~~~~~~~~
                          |
end
```

結果は次の通り。

(1)hueristic なし (freq-9.1.thr0.43.w3.nor/freq-9.1.est)
--<大計>-- 会話数: 166 句点: 289 節境界: 123
○:273 ×:16 ↓:2608 ※:57 再現率:0.944637 適合率:0.827273
正解数:2881 全語数:2954 合計正解率:0.975288 平均正解率:0.885955
~~~~~~~~ ~~~~~~~~

(2)hueristic あり (freq-9.1.thr0.43.w3.heu/freq-9.1.est)
--<大計>-- 会話数: 166 句点: 289 節境界: 123
○:337 ×:2 ↓:2564 ※:51 再現率:0.994100 適合率:0.868557
正解数:2901 全語数:2954 合計正解率:0.982058 平均正解率:0.931328
~~~~~~~~ ~~~~~~~~

※予備実験の結果について、次頁から品詞別に分類してまとめた。
<report-2.nor report-2.heu>

## ３．ＴＤＭＴで翻訳できない事例への適用

ＴＲＳの処理単位のうち、ＴＤＭＴで翻訳できないとあらかじめ報告されているものについて検討した。
結果は次頁以降参照。 <nil-TDMT.doc>

## 4．音声認識実験結果への適用

### a．音声認識結果(N-Best標準出力フォーマット)の変換

標準出力フォーマットの中のマスター単語ＩＤを参照して、 JMOR形式に変換する。

（入力）
/dept4/work3/SSSLR_PRETERM_TEST/New2LR/
NBEST/GRAMMAR/MASTER_LEXICON.1.03
HIKAKU/WG/SIALL/TXXnnnnn.si.n_best
/DB/SLDB/LNG/JMOR/TXXnnnnn.JMOR
（ツール）
nb2jm-9.csh (nb2jm.awk)
（出力）
JMOR.9.nbest/TXXnnnnn.JMOR

### b．評価のための句点付加

評価の基準になる句点は nb2jm.awk が SLDBのJMORと比較して、文の先頭から完全に一致するところまでを自動付加している。認識のゆれによっては句点の付加ができないところもある。場合によっては、ヒューリスティックの規則を変更することで対応する。

（対象ファイル）
JMOR.9.nbest/TXXnnnnn.JMOR
heuristic.grm

### c．推定頻度算出と評価

予備実験で作成したツール(freq.awk, freq-est.awk)を使って、推定頻度の算出と評価を行なう。

（入力）
JMOR.9.nbest/TXXnnnnn.JMOR
heuristic.grm
（ツール）
recog-9.csh (freq.awk, freq-est.awk)
（出力）
NBEST-9/NBEST-9.1.est
NBEST-9/TXXnnnnn.est
（まとめ）
NBEST-9/report-9.YYMMDD (YYMMDD=年月日)

※次頁からまとめ<NBEST-9/report-9.970804>

５．使用ファイル
（フルパス指定の無いファイルは/home/as54/nohtsuki/KUTEN/CLAUSE 以下）

ａ．閾値の設定

```
THRESHOLD/
  freq-9.{単語条件}.thr.w{単語組数}.nor.mat 閾値による正解値の推移表
```

ｂ．予備実験

```
JMOR.9.learn/*.JMOR          訓練セット
JMOR.9.test/*.JMOR           予備実験用テストセット
heuristic.grm                ヒューリスティックの規則

freq-9.{単語条件}.thr{閾値}.w{単語組数}.nor/
  freq-9.{単語条件}.est      ２語頻度。タスク毎、および全タスクの正解率
  TXXnnnnn.est               ４語頻度。タスク毎の正解率
  report-1.{日付}            ○(正解)について
  report-2.{日付}            ×(誤り)※(涌き出し誤り)について
```

ｃ．音声認識結果への適用

```
JMOR.9.learn/*.JMOR          訓練セット
JMOR.9.nbest/*.JMOR          音声認識結果用テストセット
heuristic.grm                ヒューリスティックの規則

NBEST-9/NBEST-9.{単語条件}.est  全体の情報
NBEST-9/TXXnnnnn.est            タスク毎の詳細情報
NBEST-9/report-9.doc            まとめ

/dept4/work3/SSSLR_PRETERM_TEST/New2LR/HIKAKU/WG/SIALL/
  TXXnnnnn.si.n_best         音声認識結果
```

ｄ．ドキュメント

```
DOC/CLAUSE.doc               このファイル
DOC/threshold.doc            閾値の分布表作成ツールについて
DOC/report.doc               品詞分類ファイルの内容について
DOC/nil-TDMT.doc             ＴＤＭＴで翻訳できない事例への適用
DOC/hueristic.doc            ヒューリスティックの規則
```

※凡例

| | 解　　説 | 実用数 |
|---|---|---|
| {単語条件} | 0:品詞のみ 1:品詞 & 活用 2:表層 & 品詞 & 活用 | 1 |
| {閾値} | 節境界の評価の基準となる頻度 | ０．４３ |
| {単語組数} | 推定頻度の算出時に組み合わせる単語数　3 or 4 | 3 |

6．使用ツール
（フルパス指定の無いファイルは/home/as54/nohtsuki/KUTEN/CLAUSE 以下）

a．閾値の設定

| | |
|---|---|
| threshold.csh | 閾値の模索の支援 |
| threshold.awk | 〃 |
| threshold-matrix.awk | 閾値別の正解率推移表を作成 |

b．予備実験

| | |
|---|---|
| freq-9.csh | 予備実験のための処理一式 |
| freq.awk | 4語頻度の算出 |
| freq-est.awk | 評価・正解率の算出 |
| report-edit.csh | まとめのための編集支援 |
| report-edit.awk | 〃 |

c．音声認識結果への適用

| | |
|---|---|
| nbest-9.csh | 音声認識結果への適用のための処理一式 |
| nb2jm-9.csh | 標準フォーマットをＪＭＯＲ形式に変換 |
| nb2jm.awk | 〃 |
| freq.awk | 4語頻度の算出 |
| freq-est.awk | 評価・正解率の算出 |

d．共通に使用する関数群

get_key.awk
split_4words.awk
heuristic.awk

---

参考文献

[1] Alon Lavie: "GLR*:A robust Grammar-Focused Parser for Spontaneously Spoken Languages," School of Computer Science, Carnegie Mellon University, CMU-CS-96-126(May 1996).

<< 閾値の決定について >>　　1997-06-26 The.

１．使用方法

```
% cd /home/as54/nohtsuki/KUTEN/CLAUSE/
% threshold.awk -v Threshold=0.43 -v Outfile=xxxx
                               ~~~~          ~~~~
                                閾値        総合情報出力先
                  freq-9.1.thr0.43.w3.heu/*.est  > /dev/null
                  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~    ~~~~~~~~~~
                            評価ファイル          タスク毎情報
                                                  出力先 default:/dev/stderr

% threshold-matrix.awk freq-9.1.thr* >threshold.mat
                       ~~~~~~~~~~~~~  ~~~~~~~~~~~~~
                      正解率ファイル      出力表
```

※threshold.awk と threshold-matrix.awk をまとめて、threshold.csh で行なう。

２．前提条件

/DB/SLDB/LNG/JMOR/*.JMOR から freq.awk と freq-est.awk を使って、TXXnnnnn.est の形式の２語頻度を記述したファイルを作成しておく必要がある。 freq-est.awk でも閾値を指定するようになっているが、ここでの値は threshold.awk には影響無い。

３．出力結果の例

```
--<大計>--
   ○:19112 ×:2632 境界正解率:0.878955  ↓:226740 ※:4967 非境界正解率:0.978817
   正解率:245852  全語数:253391   合計正解率:0.970248

 ○ : MIN 0.390057  MAX 1.000000                          -+
 × : MIN 0.000593  MAX 0.389512                           +-- 評価毎の推定頻度の
 ↓ : MIN 0.000000  MAX 0.389610                           |   最小値・最大値
 ※ : MIN 0.390037  MAX 0.974026                          -+

                     :    ○      ×      ↓      ※     -+
0.000000 ~ 0.100000 :      0    478 204905      0      |
0.100000 ~ 0.200000 :      0    565   9961      0      |
0.200000 ~ 0.300000 :      0    759   2245      0      |
0.300000 ~ 0.400000 :     83    830   9629    476      |
0.400000 ~ 0.500000 :   8087      0      0   1741      +-- 推定頻度別の分布
0.500000 ~ 0.600000 :   1518      0      0   1250      |
0.600000 ~ 0.700000 :    749      0      0    668      |
0.700000 ~ 0.800000 :    744      0      0    243      |
0.800000 ~ 0.900000 :   4743      0      0    515      |
0.900000 ~ 1.000000 :   2872      0      0     14      |
1.000000 ~ 1.100000 :    316      0      0      0     -+
```

## 4．結果の判定

threshold.csh を使用して、閾値を動かしながら閾値の最適値を模索する。
その結果は、次の表のようになった。境界正解率は句点の位置で閾値以上の頻度である割合、非境界正解率は句点以外で閾値未満の頻度である割合である。
境界正解率は閾値が大きくなるほど下がり、非境界正解率は上がる。この影響で、合計正解率は山なりの値になっている。

評価ファイル freq-9.1.thr0.10.w3.nor/TXXnnnn.est

使用ツール

```
------------------------------( threshold.csh )---------------------------------
#!/usr/local/bin/tcsh -xf
# threshold.csh
#
set LOG   = freq-9  # 9会話
set KIND  = 1       # 品詞 & 活用
set WORD  = 3       # 単語組数
set HEU   = nor     # nor(mal) | heu(ristic)
set INPUT = $LOG.$KIND.thr0.10.w$WORD.$HEU   # 2語頻度
#
# 当たりをつけるためにおおまかに模索する
foreach SHIKII (0.10 0.20 0.30 0.40 0.50 0.60 0.70 0.80 0.90)
    set OUTPUT = THRESHOLD/$LOG.$KIND.thr$SHIKII.w$WORD.$HEU
    threshold.awk -v Threshold=$SHIKII -v Outfile=$OUTPUT -v Scale=0.1 $INPUT/T*.est
 > /dev/null
end
#
# より詳細に模索する
foreach SHIKII (0.31 0.32 0.33 0.34 0.35 0.36 0.37 0.38 0.39 0.41 0.42 0.43 0.44 0.4
5 0.46 0.47 0.48 0.49)
    set OUTPUT = THRESHOLD/$LOG.$KIND.thr$SHIKII.w$WORD.$HEU
    threshold.awk -v Threshold=$SHIKII -v Outfile=$OUTPUT -v Scale=0.1 $INPUT/T*.est
 > /dev/null
end
#
threshold-matrix.awk THRESHOLD/$LOG.$KIND.thr*.w$WORD.$HEU >THRESHOLD/$LOG.$KIND.thr
.w$WORD.$HEU.mat
------------------------------( threshold.csh )---------------------------------
```

出力表 THRESHOLD/freq-9.1.thr.w4.nor.mat 9会話 品詞&活用 4語組 Heuristic無
THRESHOLD/freq-9.1.thr.w3.nor.mat 9会話 品詞&活用 3語組 Heuristic無

# 発話単位から言語処理単位への変換プログラム

1998-02-27

前回 (1997-08-29) 提出分の【統計的な言語モデルによる節境界検出ツールの作成】作業で作成した機能を音声翻訳統合実験システムに組み込むための改造である。

文法のＴＤＭＴ体系への対応、出力形式の対応、処理速度の向上、プログラムサイズの縮小を計った。

## もくじ

（フルパス指定の無いファイルは /home/as403/nohtsuki/KUTEN/CLAUSE-OPEN1997/ 以下）

# 1 改造点

## 1.1 環境ファイルの設置

ファイル名や各種条件で、変更の可能性の大きいパラメータをテキストファイルに記述するようにした。現時点では次のように設定している。

＜対象ファイル＞ PARAM/config.prm

```
# config.prm
MASTERDIR = /dept3/work22/yumi/TANIGAKI/TDMT_JMOR_SLDB_JOE_970604_N/
MASTER    = MASTER_LEXICON_DEPT3.phone
#
2WORDSDIR = PARAM                # 2語頻度ファイル Directory
2WORDS    = freq-2words.prm      # 2語頻度ファイル名
#
HEUDIR    = PARAM                # ヒューリスティック Directory
HEURISTIC = heuristic-TDMT.grm   # ヒューリスティック 規則
#
NBEST     = 9999                 # 順位 1→NBEST
MERGE     = score                # 候補の圧縮条件 score or times
KIND      = 1                    # 単語条件
WORD      = 3                    # 単語組数
SHIKII    = 0.43                 # 閾値
HEU       = heu                  # ヒューリスティック nor: なし heu: あり
#
LOG       = /dev/null            # 4語頻度のログ
```

### 1.2 2語頻度算出機能の切り離し

2語頻度は正解のＪＭＯＲの学習セットから算出するが、システムの実行の途中で学習セットを変更することはない。従って、実行する前にあらかじめ算出を行なっておいても良い。この処理を切り離すことで、処理速度の向上を計った。

- （入力）
  /dept3/work22/yumi/TANIGAKI/TDMT_JMOR_SLDB_JOE_970604_N/
  MASTER_LEXICON_DEPT3.phone　　マスター単語辞書
  JMOR.TDMT.learn/TXXnnnnn.JMOR　　学習セット (618 会話－ 9 会話)
- （ツール）
  freq-2words.awk　freq-2words.csh
- （出力）
  PARAM/freq-2words.prm 2語頻度ファイル

＜出力例＞

```
<< 2語単位での出現数集計 >>
   比較条件：品詞 & 活用形

分母
C0([ 助動詞 | 五段カ | 基本   ・ 普通名詞 ||            ]) =    11
C0([ 本動詞 | 一段 | 連用    ・ 補助動詞 | 五段カ | 連用 ]) =     8
C0([ 普通名詞 ||             ・ 係助詞 ||              ]) =  3063

分子
C1([ 助動詞 | 五段カ | た     ・ 助動詞 | 特殊ラ | 命令   ]) =    11
C1([ 補助動詞 | サ変 | た     ・ 助動詞 | 特殊サ | 基本   ]) =     2
C1([ 助動詞 | カ変 | た       ・ 助動詞 | 特殊ラ | 命令   ]) =     5


C2([ 本動詞 | 特殊サ | 基本   ・ 形容詞 | 形容詞 | 基本   ]) =     3
C2([ 助動詞 | 特殊ラ | 命令   ・ 感動詞 ||              ]) =     5
C2([ 助動詞 | 特殊サ | 基本   ・ サ変名詞 ||            ]) =    43
```

### 1.3 4語頻度算出機能の修正

パラメータや入出力書式の変更に伴い、本体の機能も修正した。

(1) 入力の変更点

- 環境ファイルの設置に対応。
- 2語頻度算出機能で保存したファイルを学習データとする。
- テストデータはＪＭＯＲではなく、Ｎ－Ｂｅｓｔ標準フォーマットから直接取得する。

(2) 出力の変更点

Ｎ－Ｂｅｓｔ標準フォーマットの形態素の要素間に、節境界予測で観測された境界を追加する。

- (入力)

  PARA/config.prm　　環境設置ファイル
  PARA/freq-2words.prm　　2語頻度ファイル
  PARA/heuristic-TDMT.grm　　ヒューリスティック規則
  /dept4/work3/SSSLR_PRETERM_TEST/New2LR/HIKAKU/WG/SIALL/
  TXXnnnnn.si.n_best　　音声認識ファイル

- (ツール)

  freq.awk freq.csh

- (出力)

  DEMO*/TXXnnnnn.si.n_best 境界を追加した音声認識ファイル

＜出力の追加例 (アンダーラインの部分) ＞

```
ORDER=1

WORDS=UTT-START/ 鈴木 + 様 / です + ね/SENT-END/SENT-START/ ご / 用件 / を / 承り / ます /UTT-END

wordids=5/10369+10025/10021+10056/2/1/10037/10328/10017/10404/10011/6

vars=1/1/1/1/1/1/1/1/1/1/1

divs=-/s+u+z+u+k+i+s+a+m+a/d+e+s+u+n+e+-///g+o/j+o+o+k+e+ng/o+-/u+k+e+t+a+m+a+w+a+r+i/m+a+s+u+-/-

times=0.000000/0.250000/0.770000/0.770000/0.770000/1.550000/1.670000/2.030000/2.230000/2.810000/
3.320000

score=144687246.000000
acoustic=149314446.000000
ngram=-4627200.000000
```

### 1.4 ヒューリスティック規則のＴＤＭＴ対応

対象となる文法がＳＬＤＢ体系からＴＤＭＴ体系に変更されているのに伴い、ヒューリスティック規則も変更した。

＜対象ファイル＞ PARAM/heuristic-TDMT.grm

## 2 作成ツールの説明

### 2.1 ＴＤＭＴ体系の日本語形態素データベースに話者毎の区切りを入れる

ＳＬＤＢ体系の JMOR ファイルには話者に関する区切りが入っていて、頻度の算出処理に使用している。しかし、ＴＤＭＴ体系の方には入っていない。ＳＬＤＢとの処理の互換性を保つため、話者が交替する時点で区切りを入れることにした。

- (ツール)
  /home/as403/nohtsuki/KUTEN/CLAUSE-OPEN1997/
  jmor-label.awk jmor-label.csh
- (使用法)

```
% jmor-label.awk -v Outdir=(出力 directory) (入力 (JMOR))
```

＜入力例＞

```
10|0010|10|10| はい | ハイ | はい | 感動詞 ||||
10|0010|20|20| こちら | コチラ | こちら | 代名詞 ||||
10|0010|30|30| ペニンシュラホテル | ペニンシュラホテル | ペニンシュラホテル | 普通名詞 ||||
10|0010|30|40| で | デ | だ | 判定詞 | 形容動詞 | 連用 ||
10|0010|30|50| ございます | ゴザイマス | ございます | 補助動詞 | 特殊サ | 基本 ||
10|0010|30|60|。 ||。 | 記号 ||||
20|0020|40|70| あした | アシタ | あした | 普通名詞 ||||
20|0020|50|80| 泊まる | トマル | 泊まる | 本動詞 | 五段ラ | 基本 ||
20|0020|60|90| こと | コト | こと | 形式名詞 ||||
20|0020|60|100| に | ニ | に | 格助詞 ||||
20|0020|70|110| なっ | ナッ | なる | 本動詞 | 五段ラ | た ||
20|0020|70|120| てる | テル | てる | 助動詞 | 一段 | 基本 ||
20|0020|80|130| 鈴木 | スズキ | 鈴木 | 普通名詞 ||||
20|0020|80|140| 直子 | ナオコ | 直子 | 普通名詞 ||||
20|0020|80|150| と | ト | と | 格助詞 ||||
20|0020|90|160| いい | イイ | いう | 本動詞 | 五段ワ | 連用 ||
20|0020|90|170| ます | マス | ます | 助動詞 | 特殊サ | 基本 ||
20|0020|90|180| が | ガ | が | 接続助詞 ||||
20|0020|90|190|。 ||。 | 記号 ||||
```

＜出力例＞

```
10|||| 話者１：|||||||                    ←－－－この行を追加
10|0010|10|10| はい | ハイ | はい | 感動詞 ||||
10|0010|20|20| こちら | コチラ | こちら | 代名詞 ||||
10|0010|30|30| ペニンシュラホテル | ペニンシュラホテル | ペニンシュラホテル | 普通名詞 ||||
10|0010|30|40| で | デ | だ | 判定詞 | 形容動詞 | 連用 ||
10|0010|30|50| ございます | ゴザイマス | ございます | 補助動詞 | 特殊サ | 基本 ||
10|0010|30|60|。 ||。 | 記号 ||||
20|||| 話者２：|||||||                    ←－－－この行を追加
20|0020|40|70| あした | アシタ | あした | 普通名詞 ||||
20|0020|50|80| 泊まる | トマル | 泊まる | 本動詞 | 五段ラ | 基本 ||
20|0020|60|90| こと | コト | こと | 形式名詞 ||||
20|0020|60|100| に | ニ | に | 格助詞 ||||
20|0020|70|110| なっ | ナッ | なる | 本動詞 | 五段ラ | た ||
20|0020|70|120| てる | テル | てる | 助動詞 | 一段 | 基本 ||
20|0020|80|130| 鈴木 | スズキ | 鈴木 | 普通名詞 ||||
20|0020|80|140| 直子 | ナオコ | 直子 | 普通名詞 ||||
20|0020|80|150| と | ト | と | 格助詞 ||||
20|0020|90|160| いい | イイ | いう | 本動詞 | 五段ワ | 連用 ||
20|0020|90|170| ます | マス | ます | 助動詞 | 特殊サ | 基本 ||
20|0020|90|180| が | ガ | が | 接続助詞 ||||
20|0020|90|190|。 ||。 | 記号 ||||
```

### 2.2 2語頻度の算出

節境界予測は、2語頻度の結果を4語頻度の計算に使用しているが、その2語頻度の結果を中間ファイルに出力する。2語頻度の算出には時間がかかるので、この処理を切り離した。

- (ツール)
  freq-2words.awk freq-2words.csh
- (使用法)

```
% freq-2words.awk -v Para=(環境File) (入力(JMOR)) >(出力File)
```

- (テスト用シェル)
  test1_freq-2words.csh 辞書のみＴＤＭＴ体系でのテスト
  test2_freq-2words.csh ＳＬＤＢ体系でのテスト
  test3_freq-2words.csh 同一単語ＩＤ並びの候補のマージ処理
  test4_freq-2words.csh ＴＤＭＴ体系でのヒューリスティックの調整
- (出力結果)
  /home/as403/nohtsuki/KUTEN/CLAUSE-OPEN1997/PARAM/freq-2words.prm

### 2.3 4語頻度の算出

節境界予測の最終段階。ここで使用している2語頻度とヒューリスティックファイル名は、環境File に記述している。

- (ツール)
  freq.awk freq.csh
- (使用法)

```
% freq.awk -v Para=(環境File) (入力(Nbest 標準)) >(出力File)
```

- (テスト用シェル)
  test1_freq.csh 辞書のみＴＤＭＴ体系でのテスト
  test2_freq.csh ＳＬＤＢ体系でのテスト
  test3_freq.csh 同一単語ＩＤ並びの候補のマージ処理
  test4_freq.csh ＴＤＭＴ体系でのヒューリスティックの調整

# 発話単位の分割または接合による言語処理単位への変換

1998-08-31

前々回（1997-08-29）から行なっている【統計的な言語モデルによる節境界検出ツールの作成】作業についての再調査、および、逆に文の途中で音声認識が切れてしまった場合の接合に関する調査を行なった。また、日英・英日双方向翻訳システムの実現に向けて英文の発話分割機能を組み込むための予備実験を行なった。

## もくじ

（フルパス指定の無いファイルは /home/as403/nohtsuki/KUTEN/CLAUSE/ 以下）

## 1　発話分割 (日本語版) に関する調査

前回提出分の節境界予測では、認識結果の形態素数を書き起こしデータとつき合わせて正解の節境界を決定していたため、認識結果の語数が合っていないと正解が導出なかった。これを書き起こしデータから直接、導出するようにして再計算した。

- (ツール)
  freq.awk　freq-est.awk
- (テスト用シェル)
  nbest-9-kakiokosi.csh
- (入力)
  JMOR.9.learn/ 学習データ JMOR.9.nbest-kakiokosi/　音声認識データをＪＭＯＲ形式にした上、書き起こし基準で句読点を付加したデータ
- (出力結果)
  NBEST-9-kakiokosi/NBEST-9-kakiokosi.1.est
  書き起こしデータ基準 (⟨⟩ 内は句点を正解に含めない場合)

```
                                |
--<大計>-- 会話数:  166  句点:  277  節境界:  111
   ○:316 ×:9 ↓:2497 ※:6  再現率:0.972308 適合率:0.981366
   正解数:2813  全語数:2828   合計正解率:0.994696 平均正解率:0.976837
<  ○:157 ×:9 ↓:2497 ※:6  再現率:0.945783 適合率:0.963190 >
<  正解数:2654  全語数:2669   合計正解率:0.994380 平均正解率:0.954487 >
```

## 2 発話接合 (日本語版) に関する調査

発話分割とは逆に、文の途中で発話が切れている場合に、次の発話と結合できる確率を調査した。音声認識でポーズ長を 300ms, 400ms, 500ms, 1000ms と設定している場合それぞれについて算出している。

- (ツール)
  freq.awk freq-est.awk
- (テスト用シェル)
  freq-9.ms.[nor—heu].csh
- (入力)
  JMOR.9.learn/ 学習データ JMOR.9.test.[pause—sent].(ポーズ長)ms/ ポーズ長に合わせて疑似的に発話単位を調整したテストデータ
- (出力結果)
  freq-9.[pause—sent].(ポーズ長)ms.nor/ 発話接合用ヒューリスティクスなし
  freq-9.[pause—sent].(ポーズ長)ms.heu/ 発話接合用ヒューリスティクスあり
  TASnnnnn.est タスク毎の結果
  *.1.est 全タスク分の算出結果
  setugou.lst 接合の文例
  setugouN.lst 非接合の文例
  setugouX.lst 句点で接合と判定された文例

←：接合（正解）、＊：非接合（誤り）、X：句点で接合（涌き出し誤り）

[ヒューリスティックなし] ※発話分割用のヒューリスティックは採用している。

300ms
--<大計>-- 会話数: 166 句点: 289 節境界: 122
←:59 ＊:29 X:11 再現率:0.670455 適合率:0.842857

400ms
--<大計>-- 会話数: 166 句点: 289 節境界: 122
←:36 ＊:17 X:10 再現率:0.679245 適合率:0.782609

500ms
--<大計>-- 会話数: 166 句点: 289 節境界: 122
←:21 ＊:10 X:10 再現率:0.677419 適合率:0.677419

1000ms
--<大計>-- 会話数: 166 句点: 289 節境界: 123
←:5 ＊:2 X:8 再現率:0.714286 適合率:0.384615

[ヒューリスティックあり]

300ms
--<大計>-- 会話数: 166 句点: 289 節境界: 122
←:59 ＊:6 X:9 再現率:0.907692 適合率:0.867647

400ms
--<大計>-- 会話数: 166 句点: 289 節境界: 122
←:36 ＊:4 X:8 再現率:0.900000 適合率:0.818182

500ms
--<大計>-- 会話数: 166 句点: 289 節境界: 122
←:21 ＊:3 X:8 再現率:0.875000 適合率:0.724138

1000ms
--<大計>-- 会話数: 166 句点: 289 節境界: 123
←:5 ＊:0 X:6 再現率:1.000000 適合率:0.454545

X（句点での接合）は体言止めや、接続助詞で止めている場合（例：～ですので。）に発生している。これらは一般的には接合する方が普通なので、発話分割の時と同様にヒューリスティックで調整できない。

ヒューリスティックを入れても←（接合）の数が変化せずに、＊（非接合）が減っているのは、感動詞（例：はい、えーっと）や、終助詞（例：～す｜か）のように句点と見なしても良いケースを○（分割）としているからである。

（フルパス指定の無いファイルは /home/as403/nohtsuki/KUTEN/CLAUSE/ 以下）

## 3 発話分割 (英語版) の予備実験

日本語の発話分割で使ったのと同じ手法で、英語の発話分割がどの程度実現できるかということを調査するための実験を行なった。

### 3.1 2語頻度の算出

まず学習セットの2語頻度の分布結果を中間ファイルに出力する。学習データとして、EMOR+(TDMT) データを使用した。形態素単位として複数単語になる場合もあるが、複数単語のまま1語として扱った。

- (ツール)
  freq-2words.awk freq-2words.csh
- (使用法)

```
% freq-2words.awk -v Para=(環境File) (入力(JMOR)) >(出力File)
```

- (入力)
  EMOR.TDMT/*.EMOR

- (出力結果)

  PARAM/freq-2words.prm

```
<< 2語単位での出現数集計 >>
   比較条件：表層 & 品詞 & 活用形

分母
C0([ reasons||PL|CN        · to|||PREP             ]) =     1
C0([ date|||CN             · size|||CN             ]) =     1
C0([ operated|PP||V        · by|||PREP             ]) =     1
C0([ while|||CONJ          · watching|ING||V       ]) =     2
C0([ suits||PL|CN          · start|||V             ]) =     1
                           |
分子   C1 = w1 w2
C1([ chinese|||CN          · medicine|||CN         ]) =     4
C1([ the|||DET             · statue of liberty|||PROPN ]) =    10
C1([ reserved|PP||V        · flight|||CN           ]) =     1
C1([ is||3S|BEV            · indoor|||ADJ          ]) =     1
C1([ going|ING||V          · over there|||LOCADV   ]) =     1
                           |
       C2 = w2 w3
C2([ osaka|||PROPN         · well|||UH             ]) =     1
C2([ hyphen|||CN           · one|||NUM             ]) =     1
C2([ las vegas|||PROPN     · could|PAST||AUXV      ]) =     1
C2([ suzuki|||PROPN        · room|||CN             ]) =     1
C2([ that|||PRON           · thank you|||INTERJ    ]) =     9
                           |
       C3 = w3 w4
C3([ but|||CONJ            · be|||BEV              ]) =     1
C3([ and|||CONJ            · here|||LOCADV         ]) =     5
C3([ and so|||CONJ         · let|||V               ]) =     1
C3([ what time|||WHPRON    · does||3S|AUXV         ]) =     8
C3([ with|||PREP           · a|||DET               ]) =     2
```

### 3.2 4語頻度の算出

学習セットから算出した2語頻度をもとに、テストセット9会話の4語頻度を算出し、節境界の判定も行なう。閾値は次項の「判定基準の閾値の探索」で検討するが、ここでは仮に 0.10 と決めて各種条件下での統計値を求める。

- (ツール)
  freq.awk freq.csh
- (テスト用シェル)
  freq-9.csh
- (入力)
  EMOR.TDMT.test/*.EMOR
- (出力結果)
  freq-9.0.thr0.10.w3.nor/ 品詞のみ・閾値 0.10・3単語
  freq-9.0.thr0.10.w4.nor/ 品詞のみ・閾値 0.10・4単語
  freq-9.1.thr0.10.w3.nor/ 品詞と活用・閾値 0.10・3単語
  freq-9.1.thr0.10.w4.nor/ 品詞と活用・閾値 0.10・4単語
  freq-9.2.thr0.10.w3.nor/ 表層と品詞と活用・閾値 0.10・3単語
  freq-9.2.thr0.10.w4.nor/ 表層と品詞と活用・閾値 0.10・4単語
  freq-9.3.thr0.10.w3.nor/ 表層のみ・閾値 0.10・3単語
  freq-9.3.thr0.10.w4.nor/ 表層のみ・閾値 0.10・4単語
    TASnnnnn.est タスク毎の結果
    freq-9.3.est 全タスク分の算出結果

### 3.3 判定基準の閾値の探索

節境界が正しい確率が最も高くなる頻度を閾値とするために探索処理を行なった。

- (ツール)
  threshold.awk threshold-matrix.awk
- (テスト用シェル)
  threshold.csh
- (入力)
  freq-9.[0-3].thr0.10.w(語数).nor/T*.est
- (出力結果)
  THRESHOLD/
  - thr-touten.mat.980625 カンマ、ピリオドともに節境界とみなした実験結果
  - thr-kuten.mat.980625 カンマを節境界の対象からはずした実験結果

# 発話単位の分割による言語処理単位への変換

1999-02-26

1997-08-29 提出分の【統計的な言語モデルによる節境界検出ツールの作成】作業で作成した機能を 1998-02-27 提出分の作業により、日本語についてはすでに音声翻訳統合実験システムに組み込んでいる。これを英語についても組み込み、日英・英日双方向翻訳システムに対応させた。

また、単語ＩＤの取得方法を変更し、日本語版・英語版を共通化させる方針で改造を行なったため、両言語とも確認を行なった。

## もくじ

（フルパス指定の無いファイルは /dept4/work4/nohtsuki/KUTEN/CLAUSE-OPEN1998/ 以下）

# 1 単体試験環境

## 1.1 実行条件

ＨＰ－ＵＸで動作確認を行なった。

## 1.2 ディレクトリ構成

```
./
    |
    |-- makefile                        メイクファイル
    |-- punc                            実行モジュール（単体試験用）
    |-- punc.c                          ソース本体
    |
    |-- sprec97_bunkatsu.c
    |-- bunkatsu_local.h
    |-- iotools.h
    |
    |-- ATRfsautil.h                    SPREC のヘッダ
    |-- ATRresult.h                     SPREC のヘッダ
    |-- ATRresult_local.h               SPREC のヘッダ
    |-- libATRbunkatsu.a                SPREC のライブラリ
    |
    |
    |-- PARAM/                          パラメータファイル
    |   |-- JE/                           日本語版
    |   |   |-- config.prm                  環境ファイル
    |   |   |-- freq-2words.prm             2語頻度ファイル
    |   |   |-- heu4.grm                    ヒューリスティック規則 (SLDB)
    |   |   +-- heu6.grm                    ヒューリスティック規則 (TDMT)
    |   |
    |   +-- EJ/                           英語版
    |       |-- config.prm                  環境ファイル
    |       +-- freq-2words.prm             2語頻度ファイル
    |
```

続く

```
|
|-- IDATA/                              入力ファイル
|   |-- JE/                               日本語版
|   |   |-- TAS12010.si.n_best              音声認識結果 (1997 年)
|   |   |-- mk_nbest.pl                       〃      (1998 年) 編集ツール
|   |   |-- example.nbest.org                 〃      (1998 年) 編集元
|   |   +-- example.nbest.cnv                 〃      (1998 年) 編集後
|   |
|   +-- EJ/                               英語版
|       |-- TAC22012.B.B60.res              音声認識結果
|       |-- TAC22012.B.B70.res                〃
|       |-- TAS12008.B.B60.res                〃
|       |-- TAS12008.B.B70.res                〃
|       |-- TAS12010.B.B60.res                〃
|       |-- TAS12010.B.B70.res                〃
|       |-- TAS22001.B.B60.res                〃
|       |-- TAS22001.B.B70.res                〃
|       |-- TAS32002.B.B60.res                〃
|       +-- TAS32002.B.B70.res                〃
|
|-- ODATA/                              出力ファイル
|   |-- JE/                               日本語版
|   |   |-- tas12010.old.log              1997 年データでの結果  punc 改造前
|   |   |-- tas12010.new.log              1997 年データでの結果  punc 改造後
|   |   |-- example.old.log               1998 年データでの結果  punc 改造前
|   |   +-- example.new.log               1998 年データでの結果  punc 改造後
|   |
|   +-- EJ/                               英語版
|       |-- tac22012.0.28.log               閾値 0.28 での結果
|       +-- tac22012.0.44.log               閾値 0.44 での結果
|
+-- SAVE/                               保存ファイル
    |-- JE/                               日本語版 (1997 年度分のファイル)
    |   |-- MASTER_LEXICON.master           マスター単語辞書
    |   |-- RECOG_LEX.vn1000                音声認識辞書
    |   |-- Makefile.bak
    |   |-- punc.c                          ソース
    |   |-- punc                            実行モジュール
    |   +-- bunkatsu_local.h
    |
    +-- EJ/                               英語版
        +-- reduced_hrt_reco_lex.1000.clerk   音声認識ファイル例
```

### 1.3 ＴＤＭＴ体系の英語形態素データベースの加工

1998年6月8日時点では、EMORのピリオドとカンマの情報が削除されていたので、ETEXTを元に、EMORのピリオドとカンマを復元した。

- （入力）
  /DB/SLDB/LNG/ETEXT+/
  TXXnnnnn.ETEXT　書き起こし情報
  /dept1/work1/ResearchE/ldata/ETDMT_NEW_980605/SLDB/
  TXXnnnnn.EMOR　ＴＤＭＴ用EMOR

- (ツール)
  /dept4/work4/nohtsuki/KUTEN/CLAUSE.E/
  emor-add.awk

- (シェル)
  emor-add.csh

- (使用法)

```
% emor-add.awk -v Outdir=(出力directory) (入力(EMOR))
```

- (出力結果)
  ./EMOR.TDMT/TXXnnnnn.EMOR　復元したEMOR

後で、次の状態に振り分けて配置する。

EMOR.TDMT.learn/TXXnnnnn.EMOR　学習セット(618会話－9会話)
EMOR.TDMT.test/TXXnnnnn.EMOR　テストセット(9会話)

### 1.4 判定基準の閾値の探索

/dept4/work4/nohtsuki/KUTEN/CLAUSE.E/
節境界が正しい確率が最も高くなる頻度を閾値とするために探索処理を行なう。

- （ツール）
  threshold.awk threshold-matrix.awk

- （シェル）
  threshold.csh 適宜、書き換えて使用する。

- （入力）
  freq-9.[0-3].thr0.10.w(語数).nor/T*.est

- （出力結果）
  THRESHOLD/
  thr-kuten.mat.980803

### 1.5 2語頻度の算出

あらかじめ2語頻度の算出を行ない、結果をファイルに出力しておく。

- (ツール)
  /dept4/work4/nohtsuki/KUTEN/CLAUSE.E/
  freq-2words.awk
- (シェル)
  freq-2words.csh 適宜、書き換えて使用する。
- (入力)
  /dept4/work4/nohtsuki/KUTEN/CLAUSE.E/
  EMOR.TDMT.learn/TXXnnnnn.EMOR 学習セット (618 会話− 9 会話)
- (出力結果)
  ./PARAM/(JE—EJ)/freq-2words.prm

### 1.6 入力ファイルの指定

修正すべき音声認識結果は標準入力から得るが、それ以外に必要なファイルは SPREC での config で設定する。ただし、単体試験環境ではプログラム (punc.c) 中に次のように記述している。

```
/*****************/
/* 単体テスト用 */
/*****************/
/* These three files should be in $SIP_SPREC97/bunkatsu/ */
#define INPUTJ1 "./PARAM/JE/heu6.grm"          /* 分割を検証するための規則 */
#define INPUTJ2 "./PARAM/JE/freq-2words.prm"   /* 分割に関する頻度情報     */
#define INPUTJ3 "/DB/SHARE/MASTER_LEX/TDMT/MASTER_LEXICON" /* マスター辞書 */
#define INPUTJ4 "./SAVE/JE/RECOG_LEX.vn1000"   /* 音声認識辞書             */
#define INPUTJ5 "./PARAM/JE/config.prm"        /* パラメータファイル       */

#define INPUTE1 "./PARAM/JE/heu6.grm" /*Dummy*//* 分割を検証するための規則 */
#define INPUTE2 "./PARAM/EJ/freq-2words.prm"   /* 分割に関する頻度情報     */
#define INPUTE3 "/DB/SHARE/MASTER_LEX/TDMT/ENGLISH_MASTER_LEXICON.txt" /* マスター辞書 */
#define INPUTE4 "./SAVE/EJ/reduced_hrt_reco_lex.1000.clerk" /* 音声認識辞書 */
#define INPUTE5 "./PARAM/EJ/config.prm"        /* パラメータファイル       */
```

### 1.7 実行モジュール (単体試験用) の作成

```
% make standalone  または  % make -DSTANDALONE punc
```

### 1.8 実行

```
% cat (音声認識結果) | punc (jap|eng) > (出力ファイル)
                        jap : 日本語版
                        eng : 英語版
```

### 1.9 出力結果

節境界を記した出力結果を以下に示す。実際は、WORDS 以下の項目は1行になっている。

```
UTTERANCE=6
amname=[/home/szhang/test/HMnet]
utime=7.920000
abstime=0.000000
cputime=36.110000
NBEST=3

ORDER=1
WORDS=UTT-START/SENT-START/okay/SENT-END/SENT-START/i/might/we/to/have/one/twin_room
/available/on/a/fifty/SENT-END/UTT-END
wordids=5/1/10047/2/1/10009/10337/10041/10024/10042/10148/10196/10194/10282/10054/10
106/2/6
vars=1/1/1/1/1/1/1/1/1/1/1/1/1/2/2/1/1/1
divs=SIL,0.650000,8383660.000000//OW,0.220000,-1150146.000000+K,0.240000,-1113714.00
0000+EY,0.430000,509735.000000+SIL,0.950000,11290736.000000///AY,0.560000,
                                    (中略)

times=0.000000/0.650000/0.650000/2.490000/2.490000/2.490000/3.050000/3.530000/3.7100
00/3.910000/4.150000/4.410000/5.110000/5.690000/6.150000/6.290000/7.920000/7.920000
score=13520012.000000 acoustic=18921012.000000 ngram=-5401000.000000
```

区切りの部分は、次の文字列が付加される。

- WORDS　/SENT-END/SENT-START/
- wordids /2/1/
- vars　/1/1/
- divs　///
- times　/(1つ前の浮動小数点数)/(1つ前の浮動小数点数)/

## 2 改造点

### 2.1 環境ファイルの設置

ファイル名や各種条件で、言語によって変更する必要のあるパラメータをテキストファイルに記述するようにした。
現時点では次のように設定している。

＜対象ファイル＞ PARAM/JE/config.prm 日本語

```
##############################
# config.prm   for Japanese
##############################
KIND      = 1                   # 単語条件 1: 品詞&活用形&活用型
WORD      = 3                   # 単語組数
SHIKII    = 0.43                # 閾値
HEU       = heu                 # ヒューリスティック nor: なし heu: あり
#
```

＜対象ファイル＞ PARAM/EJ/config.prm 英語

```
##############################
# config.prm   for English
##############################
KIND      = 2                   # 単語条件 2: 表層&品詞&活用形&活用型
WORD      = 4                   # 単語組数
SHIKII    = 0.28                # 閾値
HEU       = nor                 # ヒューリスティック nor: なし heu: あり
#
```

### 2.2 単語ＩＤの取得方法の変更

今までは単語ＩＤをマスター単語辞書から取得していたが、マスター単語辞書を全て読み込むのは無駄があるため、音声認識辞書から取得するように変更した。
ただ、ＩＤに対応する７つ組みデータは、音声認識辞書に注釈として記述されているため、記述されていない場合もあるので、マスター単語辞書から取得する。
これにより、音声認識辞書に存在するデータのみを取得することになる。

音声認識辞書のＩＤ→（マスター単語辞書のＩＤ）→マスター単語辞書の７つ組みデータ

### 2.3 ヒューリスティック規則

英語については、ヒューリスティック規則を導入していない。