NAIST Technical Report
NAIST-IS-TR99009

Japanese Morphological Analysis System ChaSen version 2.0 Manual

Yuji Matsumoto, Akira Kitauchi, Tatsuo Yamashita and
Yoshitaka Hirano

April 1999 Copyright © 1999 Nara Institute of Science and Technology.

Japanese Morphological Analysis System ChaSen Manual
Yuji Matsumoto, Akira Kitauchi, Tatsuo Yamashita and Yoshitaka Hirano
Copyright © 1999 Nara Institute of Science and Technology. All Rights Reserved.

Use, reproduction, and distribution of this software is permitted. Any copy of this software, whether in its original form or modified, must include both the above copyright notice and the following paragraphs.

Nara Institute of Science and Technology (NAIST), the copyright holders, disclaims all warranties with regard to this software, including all implied warranties of merchantability and fitness, in no event shall NAIST be liable for any special, indirect or consequential damages or any damages whatsoever resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortuous action, arising out of or in connection with the use or performance of this software.

The Japanese morphological dictionary included in this system originates from ICOT Free Software. The following conditions for ICOT Free Software applies to the morphological dictionary of the system.

Each User may also freely distribute the Program, whether in its original form or modified, to any third party or parties, PROVIDED that the provisions of Section 3 ("NO WARRANTY") will ALWAYS appear on, or be attached to, the Program, which is distributed substantially in the same form as set out herein and that such intended distribution, if actually made, will neither violate or otherwise contravene any of the laws and regulations of the countries having jurisdiction over the User or the intended distribution itself.

NO WARRANTY

The program was produced on an experimental basis in the course of the research and development conducted during the project and is provided to users as so produced on an experimental basis. Accordingly, the program is provided without any warranty whatsoever, whether express, implied, statutory or otherwise. The term "warranty" used herein includes, but is not limited to, any warranty of the quality, performance, merchantability and fitness for a particular purpose of the program and the nonexistence of any infringement or violation of any right of any third party.

Each user of the program will agree and understand, and be deemed to have agreed and understood, that there is no warranty whatsoever for the program and, accordingly, the entire risk arising from or otherwise connected with the program is assumed by the user.

Therefore, neither ICOT, the copyright holder, or any other organization that participated in or was otherwise related to the development of the program and their respective officials, directors, officers and other employees shall be held liable for any and all damages, including, without limitation, general, special, incidental and consequential damages, arising out of or otherwise in connection with the use or inability to use the program or any product, material or result produced or otherwise obtained by using the program, regardless of whether they have been advised of, or otherwise had knowledge of, the possibility of such damages at any time during the project or thereafter. Each user will be deemed to have agreed to the foregoing by his or her commencement of use of the program. The term "use" as used herein includes, but is not limited to, the use, modification, copying and distribution of the program and the production of secondary products from the program.

In the case where the program, whether in its original form or modified, was distributed or delivered to or received by a user from any person, organization or entity other than ICOT, unless it makes or grants independently of ICOT any specific warranty to the user in writing, such person, organization or entity, will also be exempted from and not be held liable to the user for any such damages as noted above as far as the program is concerned.

JUMAN	
Version 0.6	17 February 1992
Version 0.8	14 April 1992
Version 1.0	25 February 1993
Version 2.0	11 July 1994
ChaSen	
Version 1.0	19 February 1997
Version 1.5	7 July 1997
Version 1.51	29 July 1997
Version 2.0b6	20 April 1999
ChaSen for Windows95	
Version 1.0	29 March 1997

Contents

1 Introduction **1**

2 Grammar and Dictionaries **1**

3 Morphological Analysis **2**
3.1 Algorithm 2
3.2 Coping with Unknown Words 3
3.3 Unknown Connectivity Cost 3

4 Installation **3**

5 How to Use ChaSen System **4**
5.1 Running ChaSen Program 4
5.2 Options 4
5.3 ChaSen Server and Client 5
5.4 Output Format 5

6 `chasenrc` Resource File **7**

7 ChaSen Library **8**

8 Calling ChaSen from Other Languages **8**
8.1 Emacs Lisp Version of ChaSen Client 8

9 Contact **9**

1 Introduction

ChaSen is a Japanese morphological analysis system which basically has the following facilities and features.

- It segments Japanese text (sentences) string into morphemes and tags those morphemes with their parts of speech and pronunciations. It also tokenizes conjugative morphemes, i.e., it tags the conjugative morphemes with their base forms and conjugation types/forms.
- In its grammar and dictionaries, morphemes as well as connectivity of two morphemes / parts of speech are defined, where some costs are assigned to their definition.
- In its morphological analysis process, ChaSen sums up those costs of morphemes and their connectivities, then outputs results with the minimum cost.
- Basically, connectivity of two morphemes / parts of speech is defined in the form of their bi-grams. In the case of the current dictionary (ipadic1.0), connectivity of two morphemes / parts of speech and its costs are automatically extracted from a parts-of-speech tagged Japanese newspaper article corpus. In order to tune its costs, part of speech bi-gram Markov model is employed, and the probability parameters of maximum likelihood estimate (MLE) model is transformed into its connectivity costs. Similarly, costs of morphemes are also obtained from the MLE model.

2 Grammar and Dictionaries

	Morpheme Files	Grammar Files
Definition Files	Morpheme Definition Files Morpheme Dictionaries	Grammar Definition Files Parts of speech File Conjugation Types File Conjugation Forms File Connectivity Rules File
System Files	System Dictionaries Index Files	Connectivity Table Connectivity Matrix

Table 1: Grammar/Dictionary Files

As shown in Table 1, grammar and dictionary files of ChaSen system can be classified using two dimensions. According to the first dimension, they can be classified into *Definition Files* and *System Files*. Definition Files include definitions of the grammar and the morphemes of ChaSen, and are automatically compiled into System Files which are used in the morphological analysis. Using the second dimension, they can be classified into *Morpheme Files* and *Grammar Files* according to the linguistic type of the contents of the files.

The description of those grammar and dictionary files is summarized below.

1. **Definition Files**

   (a) Morpheme Definition Files

   - *Morpheme Dictionaries* (`Noun.dic`, etc.)
     define morphemes of each part of speech. A morpheme is defined as a list of its surface form (or its base form if conjugative), pronunciation, conjugation type if

conjugative, and semantic information. A surface form cost (to be used in the morphological analysis) can be assigned to each morpheme definition.

(b) Grammar Definition Files

- *Parts of speech File* (`chasen.grammar/grammar.cha`)
  defines the set of parts of speech.
- *Conjugation Types File* (`chasen.ctypes/ctypes.cha`)
  defines the set of conjugation types for each conjugative part of speech.
- *Conjugation Forms File* (`chasen.cforms/cforms.cha`)
  defines possible conjugation forms for each conjugation type.
- *Connectivity Rules File* (`chasen.connect.c/connect.cha.c`)
  defines connectivity of two morphemes / parts of speech in the form of their bi-grams. A connectivity cost has to be assigned to each bi-gram of morphemes / parts of speech.

2. **System Files**

(a) Morpheme Files

- *System Dictionaries* (`*.int`)
  is obtained by compiling morpheme dictionaries and encoding morpheme information.
- *Index Files* (`*.pat`)
  include Patricia tree indices of system dictionaries.

(b) Grammar Files

- *Connectivity Table* (`chasen.table/table.cha`)
  defines the correspondence between the rows/columns of the connectivity matrix and the morphemes / parts of speech listed in the connectivity rules file.
- *Connectivity Matrix* (`chasen.matrix/matrix.cha`)
  defines connectivity of two morphemes / parts of speech in the form of a matrix.

3 Morphological Analysis

3.1 Algorithm

For the string of the input Japanese sentence, ChaSen consults its morpheme dictionaries and records all the possible morphemes that are any sub-strings of the input string. Next, ChaSen calculates following two types of costs.

Morpheme Cost A cost that is assigned to each morpheme, and is calculated as the product of

- the cost of the corresponding part of speech (defined in the `chasenrc` resource file),
- relative weight of morpheme costs (defined in the `chasenrc` resource file),
- and the surface form cost (defined in the morpheme dictionaries).

Connectivity Cost A cost that is assigned to each bi-gram of morphemes, and is calculated as the product of

- the connectivity cost defined in the connectivity rules file,
- and the relative weight of connectivity costs (defined in the `chasenrc` resource file).

For the string of the input Japanese sentence, every possible segmentation into morpheme sequences and their parts of speech tagging is considered and sum of the above morpheme costs and their connectivity costs are calculated. Then, the results with the minimum cost are returned. Some cost width of beam search is defined in the `chasenrc` resource file, and at every position in the input string, morphological analysis results are pruned using this cost width of beam search.

3.2 Coping with Unknown Words

When ChaSen consults its morpheme dictionaries with some sub-string of the input string and can not find any morphemes, it assumes that the sub-string should be considered as a morpheme and behaves as if the sub-string were contained in its morpheme dictionaries, although the sub-string is assigned an extremely high cost compared with those morphemes existing in its morpheme dictionaries. Details of this facility of coping with unknown words are as follows:

- For hiragana (Japanese), kanji (Chinese), numbers, and symbols character types, ChaSen assumes each one character as a possible unknown morpheme that is not contained in its morpheme dictionaries. On the other hand, for other character types (katakana (foreign), (English) alphabet, etc.), ChaSen assumes the longest string each character of which is of the same character type as a possible unknown morpheme that is not contained in its morpheme dictionaries.
- Those morphemes that are not contained in the morpheme dictionaries are considered as having the *part of speech for unknown words*, which is defined in the `chasenrc` resource file.
- Those morphemes that are not contained in the morpheme dictionaries are assigned the *cost for unknown words*, which is defined in the `chasenrc` resource file.

3.3 Unknown Connectivity Cost

Basically, bi-grams of morphemes that does not match any rules listed in the connectivity rules file are not allowed in the morphological analysis results. However, users can allow those prohibited bi-grams in the morphological analysis by giving them an extremely high cost. This can be done by defining *unknown connectivity cost* in the `chasenrc` resource file (how to define the unknown connectivity cost are described in the next section).

4 Installation

Suppose `$CHASEN` be the home directory of ChaSen system. (The following describes a general procedure that is common to most distributions and dictionaries. See the `$CHASEN/README` file for the detailed procedure and description that is specific to each distribution or dictionary.)

1. Modify `$CHASEN/Makefile` as you like and type "make". This produces all the necessary object programs.
2. Type "make dic". This produces the system dictionaries. The compilation time of the current dictionary (ipadic1.0) is about 50 seconds on Pentium II (400MHz). The size of the compiled dictionary (`chadic.int` and `chadic.pat`, for ipadic1.0) is about 10Mbytes in total.
3. Type "make install" to install programs. This does not install ChaSen library and Emacs Lisp version of ChaSen client.
4. To use user specific chasenrc file, copy `$CHASEN/chasenrc` to the user's home directory as the name "`.chasenrc`" and modify `.chasenrc` so that it correctly indicates the directories for the files of grammar definition and dictionaries. See section 6 for the details.

5 How to Use ChaSen System

5.1 Running ChaSen Program

Suppose a Japanese text file "*nihongo*", which should be encoded in Japanese EUC (Extended UNIX Code) or JIS (ISO-2022-JP). Issue the following command:

`chasen` *nihongo*

The result of the morphological analysis is shown on the standard output. If your terminal has a direct input facility of Japanese characters, simply type

`chasen`

then input a Japanese sentence followed by a carrige return.

5.2 Options

There are several options:

- how to run

`-s`	start ChaSen server
`-P` *port*	specify ChaSen server's port number (use with `-s`, the default is 31000)
`-D` *host[:port]*	connect to ChaSen server
`-R`	does not read chasenrc file (use with `-D`)
`-a`	run standalone even if environment variable `CHASENSERVER` is set

- how to print ambiguous results

`-b`	print one result with the least cost (default)
`-m`	print ambiguous parts explicitly
`-p`	print all possible results independently

- output format

`-f`	print the result in a table like format (default)
`-e`	print all information of each morpheme separated by a blank
`-c`	print all information of each morpheme in internal codes
`-d`	print detailed morpheme data for Prolog.
`-v`	print detailed morpheme data for ViCha.
`-F` *format*	print morpheme data with formetted output
`-Fh`	print help of the format of `-F` option

- others

`-j`	Japanese sentence mode (assume a punctuation mark as a sentence delimiter)
`-w` *width*	specify the cost width
`-C`	use command mode
`-r` *rc_file*	use *rc_file* as a chasenrc file other than the default
`-L` *lang*	specify the language of the input text
`-h`	print the help message
`-V`	print ChaSen version number

For example, compare the default output with the results of the following.

`chasen -m -e` *nihongo*

5.3 ChaSen Server and Client

You can use ChaSen server and its client. First, type

```
chasen -s
```

to start ChaSen server.
Type

`chasen` -D*host nihongo*

('*host*' should be the hostname of ChaSen server) to run ChaSen client.

5.4 Output Format

Notes about `-F` option.

format characters:

%m	surface form (conjugated form)
%M	surface form (base form)
%y	pronunciation (conjugated form)
%Y	pronunciation (base form)
%rABC	surface form with ruby
%i	semantic information
%Ic	semantic information (if NIL, print character 'c'.)
%Pc	parts of speech (name) of all the layers of the parts of speech hierarchy, concatenated with the character 'c' (only for the v-gram version)
%Pnc	parts of speech (name) of the layers 1–n of the parts of speech hierarchy, concatenated with the character 'c' (only for the v-gram version)
%h	part of speech (code)
%H	part of speech (name)
%Hn	the part of speech (name) at the n-th layer (if NIL, the part of speech at the most specific layer) (only for the v-gram version)
%b	sub-part of speech (code)
%BB	sub-part of speech (name) (if NIL, print part of speech)
%Bc	sub-part of speech (name) (if NIL, print character 'c')
%t	conjugation type (code)
%Tc	conjugation type (name) (if NIL, print character 'c')
%f	conjugated form (code)
%Fc	conjugated form (name) (if NIL, print character 'c')
%S	the input sentence
%pi	the index of the path of the output lattice
%ps	the starting position of the morpheme at the path of the output lattice
%pe	the ending position of the morpheme at the path of the output lattice
%pc	the cost of the path of the output lattice
%ppiC	the indices of the preceding paths, concatenated with the character 'C'
%ppcC	the costs of the preceding paths, concatenated with the character 'C'
%?B/STR1/STR2	if sub-part of speech exists, STR1, otherwise, STR2
%?I/STR1/STR2	unless the semantic information is NIL and "", STR1, otherwise, STR2
%?T/STR1/STR2	if conjugative, STR1, otherwise, STR2
%?F/STR1/STR2	same as %?T/STR1/STR2
%?U/STR1/STR2	if unknown word, STR1, otherwise, STR2
%U/STR/	if unknown word, "未知語", otherwise, STR
%c	cost value of the morpheme
%%	'%'
.	specify the field width
-	specify the field width
1-9	specify the field width
\n	carrige return
\t	tab
\\	back slash
\'	single quotation mark
\"	double quotation mark

example:

"%m %y %M %h %b %t %f\n"	same as '-c' option
"%m %Y %M %H %h %B* %b %T* %t %F* %f\n"	same as '-e' option

6 chasenrc Resource File

The chasenrc resource file is used for defining various options necessary for running ChaSen morphological analysis program. As for which chasenrc resource file to be used in the morphological analysis process, the following preference order holds.

1. the one given with -r option when running ChaSen program.
2. the one given as the environmental variable CHASENRC.
3. .chasenrc file at the user's home directory.
4. the default file given as the variable RCPATH in the top level Makefile.

The following gives options that are defined in the chasenrc resource file, as well as their examples.

1. Directory of grammar files (section 2).

```
(文法ファイル /usr/local/lib/chasen/dic)
```

2. System dictionaries (section 2). The suffix '.int' has to be omitted. More than one system dictionaries can be used.

```
(PATDIC /usr/local/lib/chasen/dic/chadic
        /home/mydir/chasen/dic/mydic)
```

3. Part of speech for unknown words (section 3.2).

```
(未知語 (名詞 サ変接続))
```

4. Cost of each part of speech (section 3.1).

```
(品詞コスト
        ((*)                1)
        ((未知語)           500)
        ((名詞)             2)
        ((名詞 固有名詞)    3)
)
```

5. Relative weights of connectivity and morpheme costs (section 3.1).

```
(連接コスト重み 1)      ; defalut vaule
(形態素コスト重み 1)    ; default value
```

6. Cost width of beam search (section 3.1).

```
(コスト幅 0)    ; default value
```

7. Unknown connectivity cost. (section 3.3).

(未定義連接コスト 100)

8. Output format.

   Users can specify the output format. For example, if there is the following line in `.chasenrc`, only the surface form will be printed.

   (出力フォーマット "%m ")

   Note that `-f`, `-e`, `-c`, `-d` and `-F` command line options override the format defined in `.chasenrc`.

9. String for the beginning of the sentence.

   (BOS 文字列 "sentence:[%S]\n")

10. String for the end of the sentence.

    (EOS 文字列 "EOS\n")

11. Parts of speech for concatenated morphemes output.

    ChaSen concatenates morphemes of the same part of speech if the part of speech is among those specified for concatenated morphemes output.

    (連結品詞 (名詞 数) (記号))

12. Sentence delimiter characters.

    Users can define sentence delimiter characters that are used when the ChaSen program is called with `-j` option.

    (区切り文字 "。、,!? .,!? ")

7 ChaSen Library

You can use ChaSen library `$CHASEN/lib/libchasen.a` to put ChaSen's module into other programs.

8 Calling ChaSen from Other Languages

8.1 Emacs Lisp Version of ChaSen Client

Copy `$CHASEN/chasen/chasen.el` to the Emacs Lisp directory to install. Specify hostname and port number of ChaSen server, and describe autoloaded functions in your .emacs.

```
(setq chasen-server-host "kyusu")
(setq chasen-server-port 31234)   ; the default is 31000

(autoload 'chasen-region "chasen" "ChaSen client" t)
(autoload 'chasen-line "chasen" "ChaSen client" t)
(autoload 'chasen-highlight-class-region "chasen" "ChaSen client" t)
(autoload 'chasen-property-class-region "chasen" "ChaSen client" t)
```

9 Contact

For further information, send an email to:

chasen@cl.aist-nara.ac.jp