A Portable & Quick Japanese Parser : QJP

Masayuki KAMEDA
Information and Communication R & D Center, RICOH COMPANY, LTD.
2-3, Shin-Yokohama 3-chome, Kohoku-ku, Yokohama, 222, Japan
kameda(~ic.rdc.ricoh.co.jp

Abstract

QJP is a portable and quick software module for Japanese parsing. QJP analyzes a Japanese sentence into segmented morphemes/words with tags and a syntactic bunsetsu kakari-uke structure based on the two strategies, a) Morphological analysis based on character-types and functional-words and b) Syntactic analysis by simple treatment of structural ambiguities and ignoring semantic information. QJP is small, fast and robust, because 1) dictionary size (less than 100KB) and required memory size (260KB) are very small, 2) analysis speed is fast (more than 100 words/sec on 80486-PC), and 3) even a 100-word long sentence containing unknown words is easily processed.

Using QJP and its analysis results as a base and adding other functions for processing Japanese documents, a variety of applications can be developed on UNIX workstations or even on PCs.

1 Introduction

Natural language parser/analyser is essential for allowing advanced functions in document processing systems, such as keyword extraction to characterize a text, key-sentence extraction to abstract a document, grammatical style checker, information or knowledge retrieval, natural language understanding, natural language interface and so on. But a general purpose parser requires 1) a large dictionary database with more than several tens of thousands words, 2) advanced techniques for disambiguation and processing semantics, and 3) substantial machine resources, such as a lot of memory and high speed CPU.

In addition, users must maintain additional terms in dictionaries for specialized fields. As a result, most parsers cannot be easily used in applications and it is difficult to develop a practical parser which can be easily integrated into many applications.

We changed our viewpoint in order to design and develop an applicable and usable Japanese parser. First, we focused on the unique sets of character-types in written Japanese and constructed a very small dictionary using mainly functional words in hiragana-character. Similar approaches[1][2] were used for segmentation or preliminary morphological analysis about 20 years ago, using the transition-point between types of character sets to cue word segmentation. Second, we noticed that dealing with syntactic ambiguities creates a large processing burden and even using semantic information does little to assist syntactic analysis at the current level. So we either simplified dealing of structural ambiguities or ignored semantics to lighten the syntactic processing.

We first created a prototype of our parser[3] using AWK language, and then rewrote it [4] in C so it could be included in applications. The resulting parser, named QJP, is portable, fast and robust. It is an effective parser for many general purpose applications, despite of a dictionary size of only 5 thousand words. It can analyze a 100-word sentence on a PC in less than one second, while using less than half of a megabyte of memory. In addition, it requires no further dictionary maintenance for new terms.

In this paper we describe the QJP's analysis methods and report on its current performances.

2 Analysis Method

QJP performs two types of analysis : 1) morphological analysis to segment a sentence into part-of-speech tagged morphemes and words, 2) syntactic analysis to place words into bunsetsu-depency structure. Analysis strategies are the followings:

- The morphological analysis is achieved by expanding an earlier methods[1][2] for bunsetsu or word segmentation using character-types thus allowing the use of a very small dictionary.
- The syntactic analysis uses no semantic information, only part-of-speech and other syntactic information. In addition, rather than creating all possible, or some preferable, parses, we construct the best syntactic structure preserving local ambiguities.

1Bunsetsu(文節) is a kind of phrasal unit in Japanese, consisting of one content word(内容語/概念語) such as noun(名詞), verb-noun(動名詞), verb(動詞), adjective(形容詞), verb-adjective(形容動詞) and adverb(助詞) and successive adjectives(形容語) such as auxiliary verb(助動詞) and post-positional particle(助詞), and carrying one concept.
2.1 Morphological Analysis

Characteristics of written Japanese

A Japanese sentence has no spaces between words.[Figure 1]. So it is difficult to segment a sentence into words. However, the fact that at least four distinct sets of characters [for example, kanji(日", "木", "語", etc), hiragana(の", "に", "た", etc), katakana(ノ", "ヨ", "ヲ", etc) and other characters (alphabets, numbers, symbols etc)] are used to write Japanese can be used for segmenting words. Most words written in kanji or katakana are content words, such as nouns, verb-noun and stems (語幹) of verbs or adjectives. Most words in hiragana are functional words (機能語), such as postpositional particles, auxiliary verbs and inflective suffix (活用語尾) of verbs and others [Table 1]. And the vocabulary of content words is much larger than that of functional words.

Sharing of Morphemes by Dictionary and Rules

Our strategy is that all functional words, which are few in number, are stored in the dictionary and most content words or their stems in kanji or katakana are to be extracted and given their part-of-speech candidates based on character-types.

Standard morphological analyser uses a dictionary to obtain morpheme or word candidates. But in our approach, morpheme candidates are extracted either from the morpheme dictionary or using allocation rules based on character-type. For example, if the dictionary look-up fails, the allocation rules extract each sequence of character in which all the characters belong to the same character set. Then, using the allocation rules, part-of-speech candidates are assigned based on the sequence's character set and length. The candidates are disambiguated by checking connection with the the following morphemes based on the connection table between morpheme parts-of-speech. The following morphemes, in most cases, are functional words or inflective suffixes.

The dictionary contains functional words such as postpositional particles, auxiliary verbs, formal nouns (形式名詞), adverbial nouns (副詞名詞), conjunctions (接続詞), adverbs and so on, inflective suffixes and exceptional content words which cannot be or are not covered by the allocation rules.

Here are some examples of the allocation rules for 1) 1-kanji character sequence, 2) 2-kanji character sequence and 3) katakana character sequence.

2 In this analysis, a inflected word is treated as two or more morphemes - a stem part and one or more inflection part.
Morphological Output from QJP

An example of segmented morphemes with

**Morpheme-Tags**: The example shown in Figure 2, where 8 nouns

"日本語," "外来語," etc. and 2 stems of "切れ目", "空き"

mark by "[j]", in *kana* character are recognized using allocation rules and connection table.

The words with part-of-speech tags and morpheme-divisions("","+"") are shown in Figure 3, where a compound noun "切れ目" (the 7th word) is a compound of the morphemes 8-10 ["切","空"] of "切れ目" (stem of "切れ目"

**Fig. 2 Example Japanese Sentence 1**: In English, in processing a sentence of an agglutinative language, in which divisions are not placed between words, morphological analysis is the first barrier.

**Fig. 3 Example Japanese Sentence 2**: In English, in processing a sentence of an agglutinative language, in which divisions are not placed between words, morphological analysis is the first barrier.

**Simple Treatment of Structural Ambiguities**

Structural ambiguities are usually dealt with either by generating all possible structures or by selecting the more preferable ones based on some scoring scheme. Such method usually leads to combinatorial explosions which causes a lot of memory and processing time.

For this problem we have already proposed a substitutional light method in *kakari-uke* analysis. This method extracts all possible *kakari-uke* pairs, and then rather than generate not all or some possible sets of pairs, only one best set of pairs is generated while still retaining all other possible pairs. Thus, instead of generating multiple number of sets, a most-likely set is selected and the application/user is presented with alternative *kakari-uke* pairs at the same time that the selected pairs are presented. If the application/user corrects any alternative *kakari-uke* pairs, the most-likely set is re-calculated using re-connection analysis.

To this end, semantic information must be stored for less machine resources.

**Not Using of Semantic Informations**

Most methods for analyzing Japanese use case patterns with semantic features for preference selections. However, such analysis techniques using semantic informations are not yet adequate and sometimes actually lead to adverse results.

In addition, semantic information must be stored.
in the dictionary. This reduces the merit of the very small dictionary achieved in morphological analysis section. We limit the information to morphological/word and syntactic levels [such as the presence of comma, adverbial noun, surface or syntactic similarity] without using semantic information for structural analysis.

Flow of QJP's Syntactic Analysis

Under these approaches, QJP's syntactic analyzer processes words sequence in three steps (Figure 4) each following its own set of rules. First it determines bunsetsu features [A] for each bunsetsu according to its word constituents. Second it extracts all possible kakari-uke bunsetsu pairs [marked by ' ○' in B] based on specific combinations of bunsetsu features for each bunsetsu pair. Last, it selects the best uke-bunsetsu (modifie)

[Segmentation of Words by Morphological Analyser]

| [1] | bunsetsu features |
|-----|------------------|
| [1] | Japanese language |
| [2] | Single space |
| [3] | Cutting edges |
| [4] | Not in place |
| [5] | No punctuation |
| [6] | Sentence |
| [7] | Processing |
| [8] | First utterance |

[Extraction of Possible Kakari-Uke Bunsetsu Pairs]

| [1] | [2] | [3] | [4] | [5] | [6] | [7] | [8] | [9] | [10] |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| [1] | Japanese language |
| [2] | Single space |
| [3] | Cutting edges |
| [4] | Not in place |
| [5] | No punctuation |
| [6] | Sentence |
| [7] | Processing |
| [8] | First utterance |

[Selection of Best Set of Kakari-Uke Bunsetsu Pairs]

| [1] | [2] | [3] | [4] | [5] | [6] | [7] | [8] | [9] | [10] |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| [1] | Japanese language |
| [2] | Single space |
| [3] | Cutting edges |
| [4] | Not in place |
| [5] | No punctuation |
| [6] | Sentence |
| [7] | Processing |
| [8] | First utterance |

Figure 4. Flow of Syntactic Kakari-Uke Analysis

Figure 5. Kakari-Uke Pairs with Tags
3 QJP

3.1 Implemented Software

QJP currently is implemented in the C language as both a QJP library and an interactive/batch console application, QJP workbench. They have been implemented on DOS/PC and UNIX/Sun workstation.

QJP’s dictionary consists of 4 files whose total size is less than 50KB and which contain about 3 thousand morphemes. QJP requires another control table file for the compressed 533x533 morpheme-POS connection table, the table for the allocation rules, the dictionary file indices and others, which is at most 35KB. 6 sets of morphological rules and 4 sets of syntactical rules [Table 2] are embedded in the form of if-then rule in C functions. The size of the workbench execution file on DOS is about 185KB. The total size ( executables and dictionaries) is much less than 300KB/DOS which is quite small and portable as a natural language analyser.

3.2 Analysis Experiment

QJP performances were measured for the QJP workbench using two sentence test-sets: 1 [241 sentences, average length 24.1 words/sentence] and 2 [210 sentences, average length 29.5 words/sentence].

Execution Performance

About 260KB of memory are required on DOS and 500KB on UNIX. With this amount of memory QJP can process a very long sentence, such as 100-word sentence [Figure 6].

The analysis speed is 80 to 150 words/sec on an 80486/23MHz PC and 700 to 800 words/sec on a Sun-Sparc 20. A 100-word sentence can be analyzed in less than 1 second on PC. Figure 8 shows the relationship of processing time to sentence length. Syntactic processing time is on the order of the square of the sentence length. But its coefficient is so small that the total processing time increases linearly in the range of actual long sentences.

Table 2. Linguistic data and Rules

| Syntactical Analyser | Example Sentence 1 | Example Sentence 2 |
|----------------------|--------------------|--------------------|
| Dictionary           | - | - |
| morpheme/word         | - | - |
| Dictionary            | - | - |
| Part-of-Speech        | - | - |
| Table                 | - | - |
| connection-rule       | - | - |
| source-rule           | - | - |
| word-rule             | - | - |
| feature               | - | - |
| POS                   | - | - |
| word-constituent rule | - | - |
| feature               | - | - |
| auxiliary func. verb   | - | - |
| rule                  | - | - |
| POS                   | - | - |
| feature               | - | - |
| word                    | - | - |
| POS                   | - | - |
| feature               | - | - |
| auxiliary func. verb   | - | - |
| rule                  | - | - |
| POS                   | - | - |
| feature               | - | - |
| word                    | - | - |
| POS                   | - | - |
| feature               | - | - |
| auxiliary func. verb   | - | - |
| rule                  | - | - |
| POS                   | - | - |
| feature               | - | - |
| word                    | - | - |
| POS                   | - | - |
| feature               | - | - |
| auxiliary func. verb   | - | - |
| rule                  | - | - |
| POS                   | - | - |
| feature               | - | - |
| word                    | - | - |
| POS                   | - | - |
| feature               | - | - |
| auxiliary func. verb   | - | - |
| rule                  | - | - |
| POS                   | - | - |
| feature               | - | - |
| word                    | - | - |
| POS                   | - | - |
| feature               | - | - |
| auxiliary func. verb   | - | - |
| rule                  | - | - |
| POS                   | - | - |
| feature               | - | - |
| word                    | - | - |
| POS                   | - | - |
| feature               | - | - |
| auxiliary func. verb   | - | - |
| rule                  | - | - |
| POS                   | - | - |
| feature               | - | - |
| word                    | - | - |
| POS                   | - | - |
| feature               | - | - |
| auxiliary func. verb   | - | - |
| rule                  | - | - |
| POS                   | - | - |
| feature               | - | - |
| word                    | - | - |
| POS                   | - | - |
| feature               | - | - |
| auxiliary func. verb   | - | - |
| rule                  | - | - |
| POS                   | - | - |
| feature               | - | - |
| word                    | - | - |
| POS                   | - | - |
| feature               | - | - |
| auxiliary func. verb   | - | - |
| rule                  | - | - |
| POS                   | - | - |
| feature               | - | - |
| word                    | - | - |
| POS                   | - | - |
| feature               | - | - |
| auxiliary func. verb   | - | - |
| rule                  | - | - |
| POS                   | - | - |
| feature               | - | - |
| word                    | - | - |
| POS                   | - | - |
| feature               | - | - |
| auxiliary func. verb   | - | - |
| rule                  | - | - |
| POS                   | - | - |
| feature               | - | - |
| word                    | - | - |
| POS                   | - | - |
| feature               | - | - |
| auxiliary func. verb   | - | - |
| rule                  | - | - |
| POS                   | - | - |
| feature               | - | - |
| word                    | - | - |
| POS                   | - | - |
| feature               | - | - |
| auxiliary func. verb   | - | - |
| rule                  | - | - |
| POS                   | - | - |
| feature               | - | - |
| word                    | - | - |
| POS                   | - | - |
| feature               | - | - |
| auxiliary func. verb   | - | - |
| rule                  | - | - |
| POS                   | - | - |
| feature               | - | - |
| word                    | - | - |
| POS                   | - | - |
| feature               | - | - |
| auxiliary func. verb   | - | - |
| rule                  | - | - |
| POS                   | - | - |
| feature               | - | - |
| word                    | - | - |
| POS                   | - | - |
| feature               | - | - |
| auxiliary func. verb   | - | - |
| rule                  | - | - |
| POS                   | - | - |
Analysis Performance

We used test-sets 1 and 2 for tuning and blind-test, respectively. For test-sets 1 and 2, the accuracy of analyzed morphemes/words is 99.3/99.3% and 95.7/96.1%, the accuracy of analyzed uke bunsetsus for each bunsetsu except the last one is 95.1% and 90.5%, and the accuracy of set of kakari-uke pairs in a sentence is 71.0% and 43.8%, respectively.

For sentences which have lengths of 3 to 15-bunsetsus and are morphologically analyzed correctly, the accuracy of analyzed uke bunsetsus for each bunsetsu is 97.3% and 93.6%, and the accuracy for sets of kakari-uke pairs in a sentence is 82.9% and 70.5%, respectively.

Comparison

There are no public data for the performance of other Japanese analysers, so comparison is difficult. But not only the size of files but also the performance figures for memory and speed of QJP are thought to be more than ten times better than those of existing Japanese analysers[4]. As for analysis accuracy, the morphological accuracy is a little lower than that of the existing Japanese morphological analysers using large scale dictionaries, but the syntactic analysis accuracy is thought to be no worse than that of the existing Japanese syntactic analysers.

4 Conclusions

We have designed and implemented QJP for the purpose of readily and easily applicable morphological and syntactic analyser for Japanese. The design strategies are based on 1) the morphological analysis based on character-types and functional words to reduce the size of dictionary, and 2) the syntactic analysis by simple treatment of structural ambiguities and ignoring semantic information to lighten processing.

QJP, as implemented, is portable, quick and robust. All files needed for execution including dictionary total less than 300KB on DOS. Even on a slow PC a 100-word sentence can be analyzed in less than 1 second using a small amount of memory. This performance is thought to be quite excellent. The analysis accuracy is comparable to that of other existing analysers. No dictionary maintenance is necessary for new terms.

The functions of QJP are implemented as a QJP library and a QJP workbench. We have already utilized QJP for keyword extraction, natural language query and text reading support functions[9] and are planning further applications, such as information retrieval system. Others use QJP functions for other purposes, such as linguistic data extraction.

QJP currently doesn’t segment compound kana words of Chinese-origin and leaves this segmentation to the application. In the future, we plan to real-ize such a segmenting function using on statistical data[10] and affixes[2].

References

[1] Yoshiyuki SAKAMOTO: An Automatic Segmentation for Japanese Text, Mathematical Linguistics, Vol.11, No.6, 1978 (in Japanese).
[2] Makoto NAGAO, Jun-ichi TSUJII, Akira YAMAGAMI, Shuji TATEBE: Data-Structure of a Large Japanese Dictionary and Morphological Analysis by Using It, Journal of Information Processing Society of Japan, Vol.10 No.6, pp.514-521, 1978 (in Japanese).
[3] Masayuki KAMEDA: A Quick Japanese Parser, Information Processing Society of Japan SIG Notes, 94-NL-4, 1993 (in Japanese).
[4] Masayuki KAMEDA: A Portable & Quick Japanese Processing Tool : QJP, Proc. of the 1st Annual Meeting of the Association for Natural Language Processing, 1995 (in Japanese).
[5] Masayuki KAMEDA, Shin ISHII, Hideo ITO: Interactive Disambiguation in a Japanese Analysis System, Information Processing Society of Japan SIG Notes, 84-NL-2, 1991 (in Japanese).
[6] 浅沼悟, 村松明宏, 田川通正: 強引な文法表現に対する文法的表記の効果の測定, DBI 電子化辞書利用シンポジウム論文集, 1995 (in Japanese).
[7] Sadao KUROHASHI, Makoto NAGAO: A Method for Analyzing Conjunctive Structures in Japanese, Transactions of Information Processing Society of Japan, Vol.33 No.8, pp.1022-1031, 1992 (in Japanese).
[8] Sadao KUROHASHI, Toshihisa NAKAMURA, Yuki MATSUMOTO and Makoto NAGAO: Improvement of Japanese Morphological Analyser JUMAN, Proc. of the International Workshop on Shareable Natural Language Resources, pp.22-28, 1994.
[9] Masayuki KAMEDA: Support functions for Reading Japanese text, Information Processing Society of Japan SIG Notes, 110-NL-9, 1995 (in Japanese).
[10] Koichi TAKEDA, Terumunehke FUJISAKI: Automatic Decomposition of Kana Compound Words Using Statistical Estimation, Transactions of Information Processing Society of Japan, Vol.28 No.5, pp.952-961, 1987 (in Japanese).

Example Sentences

1. Tohru HISAMITSU: Proc.of the 42th Meeting of Information Processing Society of Japan, 1991.
2. Japanese Laid Open Patent No.60-200358, 1985.