RA-009

日本語プログラミング言語における字句解析
A Lexical Analyzer for a Japanese Programming Language

馬場 祐人[†]　筧 捷彦[‡]
BAMBA Yuto　KAKEHI Katsuhiko

1. はじめに

日本語プログラミング言語は，変数や関数を日本語で名付け，また日本語の表記および語順に近い文法でプログラムを書くプログラミング言語である。日本語の文は，単語ごとにスペースや読点“、”で分かち書きしない特徴がある。このような日本語の文の特徴を活かした日本語プログラミング言語のパーサを，既存の字句解析器ジェネレータを利用して作ることは難しい。一方，自然言語処理では，形態素解析によって，日本語の文を解析する。形態素解析では，あらかじめ用意された辞書を使って，日本語の文を品詞ごとに形態素として分解する。日本語プログラミング言語で書かれたプログラムの解析においても，字句解析の代わりに形態素解析を使うことが考えられる。形態素解析には膨大な量の辞書が必要であるが，日本語プログラミング言語で書かれた日本語(プログラム言語な日本語)の文で使われる品詞や語彙は，一般の自然言語処理で対象とする日本語(自然言語な日本語)の文よりも限られる。そのため，自然言語な日本語の文の解析よりも，簡単な手法と少ない辞書でプログラム言語な日本語の文を解析することができると考えられる。

本論文では，基本的な形態素解析の手法と必要最低限の辞書を用いて，分かち書きされないプログラム言語な日本語の文を字句分割するための字句解析器の実現方法を述べる。また，この方法を実装した字句解析器を使って，筆者らが開発している日本語プログラミング言語“プロデル[1]”で書かれたいくつかのプログラムを字句解析し，その過程で挙げられた問題点と対処について述べる。

2. 背景

2.1. 既存の字句解析の問題点

日本語の文は，単語ごとにスペースや読点で分かち書きしない特徴を踏まえ，日本語プログラミング言語でも，プログラム文を自然な日本語の表記および語順に近づけて書くことができるように設計されている。日本語プログラミング言語で書かれたプログラム(日本語プログラム)は，C や Java などの一般的なプログラム言語のように，スペースや改行，記号で確実に字句が区切られることはない。そのため，Lex や Flex といった既存の字句解析器ジェネレータを利用して，明確な区切りがない日本語の文で書かれたプログラムを解析するパーサを作ることは難しい。例えば，**“もし値が 10 なら値に 1 を加える”**というプログラム言語な日本語の文があるとする。この文には，スペースや読点は含まれない。日本語プログラミング言語の処理系は，この文から**“もし**(予約語)” **“値**(変数名)” **“が**(助詞)”・・・などと，字句を切り出していく。**“もし”**のようにプログラムの制御文で使われる語であれば，字句解析の際に，その語を予約語として宣言すれば切り出すことは可能である。しかし，変数や関数，型などを適切な位置で切り出すには，解析対象のプログラムで使われる可能性のあるすべての語を事前に字句定義として宣言しておかなければならない。これは，現実的な方法とは言えない。字句解析器に切り出す語(変数や関数，型など)を動的に指定できる辞書を持たせる方法が考えられる。しかし，既存の字句解析器ジェネレータで，このような仕組みを実現できない。

2.2. 形態素解析の問題点

一方，自然言語処理分野では，日本語の文を解析する手段として形態素解析がある。形態素解析では，あらかじめ用意された辞書を使って，日本語の文を品詞ごとに形態素として分解する。形態素解析では，入力となる日本語の文がスペースや読点で区切られている必要はない。プログラム言語な日本語で書かれたプログラムにおいても，字句解析の代わりに形態素解析を使うことが考えられる。しかし形態素解析では，日本語で使われるすべての品詞と語彙を辞書に持つ必要がある。このため言語処理系に膨大な量の辞書を持たせる必要がある。例えば MeCab の場合，辞書および実行ファイルのファイルサイズは，98MB 程度である。一方，Java のコンパイル環境(JDK5)にあるコンパイラは，数 MB 程度であり，ライブラリを含めても，90MB 程度である。MeCab と同程度の辞書をコンパイラで利用することを考えると，コンパイラのサイズだけで Java 環境(コンパイラおよびクラスライブラリ)と同程度になる。実用面から考えて，日本語プログラミング言語の環境も Java の環境と同程度のファイルサイズになることが理想だと考えられる。また直感的に考えて，プログラム言語として使われる品詞や語彙は，自然言語な日本語よりも少ないはずであり，必要最低限の仕組みで言語処理系に実装できることが望ましい。

[†] 早稲田大学 理工学術院 基幹理工学研究科 Graduate School of Fundamental Science and Engineering, Waseda University
[‡] 早稲田大学 基幹理工学部 Faculty of Science and Engineering, Waseda University

3. プログラムとしての日本語

3.1. 日本語プログラムの構文

プログラム言語な日本語の字句解析について述べる前に，既存の日本語プログラミング言語のプログラムの例を挙げて，プログラム言語な日本語の特徴について考える。ここでは既存の日本語プログラミング言語として，プロデルのプログラム例を挙げる。リスト 1は，プロデルにおける関数呼出し文のプログラム例と，それに相当するプログラムを Java で書いたものである。なお，この論文では，単語の切れ目を“|”で示してある。

リスト 1　関数呼出し文の例

Java:

```
コピーする("文章.txt",　バックアップ先);
```

プロデル:

```
「文章.txt」|を|バックアップ先|へ|コピーする|。
```

リスト 1の Java のプログラムで示すように，一般的に関数呼出し文は，関数名および実引数(必要な場合)を書く。プロデルの関数呼出しも同様に関数名と実引数を書くが，Java と異なる点として，実引数と仮引数を結びつけるために“助詞”を書く点がある。実引数の直後に添えられた助詞によって，実引数と仮引数が対応づけられる。これによって，補語(実引数+助詞)[6][7]の順番に関係なく，関数へ正しく引数を渡すことができる。なお，“「”と“」”で囲まれた部分は，文字列定数である。文字列定数の途中で字句が区切られることはない。改行または“。”は，文の終わりとして扱われる。またスペースや読点は，単語の区切りとして扱われる。次に制御文が含まれるプログラム例(リスト 2)を示す。

リスト 2　制御文の例

Java:

```
if (値 == 10) 値++;
```

プロデル:

```
もし|値|が|10|なら|値|に|1|を|足す|。
```

リスト 2にある，**“もし”**および**“なら”**は，予約語である。また**“値”**は，変数である。さらに**“が”**，**“に”**および**“を”**が助詞であり，**“足す”**が関数である。リスト 1およびリスト 2を見るように，日本語プログラミング言語で用いられるプログラム言語な日本語の文は，複雑な構造ではなく，補語(名詞+助詞)と動詞で構成されている単純な文がほとんどであることが分かる。

3.2. 日本語プログラムで使う品詞

プログラム言語な日本語で使われる品詞について考える。日本語の文の構造からすると，関数にあたる部分の品詞は，動詞である。また，実引数にあたる部分は名詞であると分類できる。さらに実引数と仮引数を対応付けるための助詞が使われている。また品詞ではないが，記号も分類できる。つまりプログラム言語な日本語の文を字句解析において，名詞，助詞，動詞および記号に分類できればよい。なお，接続詞や感動詞，助動詞，形容詞などは，予約語として使われるものに限られ，プログラム言語な日本語の文では使われない。プロデルの他，現時点での日本語プログラミング言語である“TTSneo[3]”や“なでしこ[4]”，“言霊[5]”でも，使われる品詞に大きな違いはない。

このようにプログラム言語な日本語の文で使われる品詞のほとんどは，名詞，助詞および動詞に限られる。日本語プログラミング言語における字句解析で，プログラム言語な日本語の文を，これらの品詞に分類しさえすればプログラム言語で使われる多くの文を解釈することができると考えられる。

3.3. 日本語プログラムで使う語彙

プログラム言語な日本語の文で使われる語彙には，予約語，型，関数，変数が挙げられる。これらは，プログラム言語で予約されたもの，プログラム中で宣言されたもの，あらかじめ外部ライブラリで宣言されたもののいずれかである。なお外部ライブラリとは，対象プログラムの解析時に処理系へ指定したコンパイル済みのプログラムライブラリである。

自然言語な日本語の文の解析では，日本語で使われるすべての語彙を持つ必要があるが，プログラム言語な日本語の文の解析で使われる語彙は，それらよりも少ない量で解析できると考えられる。

3.4. 日本語プログラムの字句

このようにプログラム言語な日本語(日本語プログラミング言語で書かれた日本語)の文は，自然言語な日本語(一般の自然言語処理で対象とする日本語)の文より，品詞も語彙も限られる。日本語プログラミング言語の字句解析で，3.2 で述べた，名詞，助詞，動詞および記号を字句として切り出すことができれば，プログラム言語な日本語の文を解釈することができると考えられる。

4. 日本語プログラミング言語における字句解析

基本的な形態素解析の手法と必要最低限の辞書を用いて，プログラム言語な日本語の文を字句解析する方法について述べる。日本語プログラムで使われる可能性のあるすべての語を字句定義として宣言しておかなければならない問題を解決するために，既存の字句解析による字句分割に加え，形態素解析で用いられる最長一致法による字句分割を字句解析に利用する。また字句解析器には，字句分割に必要な

辞書を持たせる。辞書には，語彙とそれに対応する品詞の対を登録する(辞書に登録する語彙は後で述べる)。そしてこの辞書で最長一致する字句を，対象の日本語プログラムから順番に切り出す。ただし，この辞書に登録されていない字句が表れた場合は，その字句を未知語として扱う。また，本研究の字句解析の方法では，一般的な形態素解析と違い，単語連接についての判断は行わない。プログラム言語な日本語の文で使われる品詞が少ないことから，単語の順が正しいかどうかは，構文解析で判定する。そのため辞書に単語連接規則を登録する必要はない。

次に，字句解析器が持つ辞書にどのような語彙を登録するかについて述べる。辞書へ登録する語彙の品詞は，名詞，助詞および動詞のいずれかである。

名詞にあたる語彙には，予約語，型，変数がある。予約語は，プログラミング言語の言語仕様として定義されているものを辞書に登録する。型と変数は，外部ライブラリ上で定義された型や変数を登録する。

助詞にあたる語彙は，リスト 3に示すものに固定する。これらの助詞を辞書に登録する。なお，これらはプロデルやなでしこなどのライブラリで用意されている関数で使われている助詞を基に決めた。

リスト 3 プログラム言語な日本語で使う助詞

```
助詞::=を | へ | に | という | で | から |
        まで | は | が | として | だけ | の
```

動詞にあたる語彙には，関数や予約語がある。関数は，外部ライブラリで定義された関数を登録する。予約語は，プログラミング言語の言語仕様として定義されているものである。

字句解析の対象となるプログラムで宣言された型，関数および変数は，未知語として扱われる。これは，型，関数および変数の宣言が構文解析以降で処理されるため，字句解析の時点でこれらの型，関数および変数が辞書に登録されていないからである。

表 1は，リスト 4のプログラムを字句解析した結果を表したものである。なお，字句解析器の辞書には，関数(“**割った余り**”および“**出力する**”)が外部で定義されているものとする。表 1 で見るように，未知語となっている字句があるが，不自然な位置で単語が区切られることはない。

このように，プログラム言語な日本語の文で使われる品詞が少ないことを活かし，必要最低限の形態素解析の仕組みだけで，分かち書きのないプログラム言語な日本語の文の字句解析を実現する。

5. 実装

4章で示した字句解析の方法を実装した字句解析器を生成する字句解析器ジェネレータを作成した。この字句解析器ジェネレータは，字句定義データからC#のコードを出力する。これにより字句解析器を生成し，それによって，既存の日本語プログラミング言語のプログラムを字句解析させる。

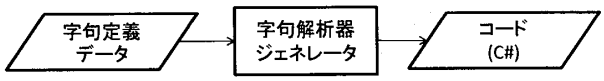

図 1　実装した字句解析ジェネレータ

リスト 4 プログラム言語な日本語の文

```
値は、変数である
値を 10 で割った余りを出力する
```

字句	分類
値	未知語
は	助詞
、	記号
変数	型
である	予約語
¥n	記号
値	未知語
を	助詞
10	数値
で	助詞
割った余り	関数
を	助詞
出力する	関数

表 1 字句解析結果

6. 評価

字句解析器ジェネレータによって生成した字句解析器を使い，プロデルで書かれたいくつかの日本語プログラムを字句解析した。その解析結果から，この論文で示した字句解析の方法の評価と，この方法で起こる問題点と対処について述べる。

6.1. 事例による評価

この論文では，例としてプロデルでリンクリストを書いたプログラム(リスト 5)を字句解析した結果を挙げる。なお，理解のために字句の切れ目を“|”で示す。また，助詞を下線，予約語を太字で示した。リスト 5について言えば，プログラム言語な日本語の文を正しい位置で字句を区切ることができた。4章で示した品詞を名詞，助詞，動詞および記号に限って字句解析する方法で，日本語プログラムを字句解析できると言える。

6.2. 字句の一部に助詞が含まれる名詞や動詞

いくつかの日本語プログラムを字句解析する過程で挙がった問題点について考える。

4章で挙げた助詞によって字句を区切ることとすると，型や変数に“の”や“に”が含まれる名前を付けた場合に，字句解析で誤った位置で区切られてしまう場合がある。例えば，次のようなプログラム言語な日本語の文である。

木の葉を集める

“**木の葉**”は，変数であるとする。この文は，“**木の葉**|**を**|**集める**”と字句分割されるべきである。この文の場合，“**木の葉**”という字句が事前に辞書に登録されていれば，正しく字句解析できる。しかし，“**木の葉**”という語が，解析処理中のプログラム内で定義されているとすると，“**木**|**の**|**葉**|**を**|**集める**”と分割される。この理由は，構文解析段階で“**木の葉**”という語の宣言が処理されるまで，この字句に含まれる“**の**”が助詞して扱われ，区切られてしまうからである。このように字句の一部に助詞が含まれる語を正しい位置で字句分割するためには，構文解析後に再度，字句解析する必要がある。つまり初回の字句解析結果をもとに構文解析を行い，新たに定義された型や変数，関数を辞書に加えて，二度目の字句解析を行う必要があると考えられる。

6.3. 字句分割の曖昧さ

プログラムで使われる単語によっては，字句の区切り方が曖昧になることがある。例えば，次に挙げるようなプログラム言語な日本語の文がある。

にわににわとりを放つ

この場合“**にわ**|**に**|**にわとり**|**を**|**放つ**”と区切られるべきであるが，“**に**|**わ**|**に**|**に**|**わとり**|**を**|**放つ**”などといくつかの種類に区切ることができる可能性がある。このように字句分割に曖昧さが生じる場合は，処理系は構文エラーとする。この場合，プログラムは，曖昧さを回避するために必要に応じて，単語を“（”と“）”で囲むか読点で単語を区切って分かち書きすればよい。上の例を記号によって分かち書きした例を，次に示す。

（にわ）に、（にわとり）を放つ

対象とする日本語の文は，プログラム言語な日本語の文であり，あくまでもプログラム言語である。そのことから，字句分割に曖昧性が生じる場合は，言語処理系がその旨を示し，プログラマに曖昧性のない書き方を求めればよいと考えられる。

7. まとめ

この論文では，形態素解析の基本的な手法である最長一致法を利用し辞書の語彙や品詞を限定することで，日本語プログラミング言語で書かれた分かち書きされない日本語の文の字句解析を言語処理系で容易に実現するための方法について述べた。またその方法で起こる問題点とその対処について述べた。分かち書きされない日本語の文で書かれたプログラムの解析の問題点の解決には，字句解析と構文解析を同時に考える必要がある。今後は構文解析についても掘り進めていく必要があると考える。

リスト 5 字句解析の例(リンクリストのプログラム)

```
//リンクリスト|
|先頭|という|リンク項目|を|作る|
|
|新項目1|という|リンク項目|を|作る|
|新項目1|の|値|を|「A」|とする|
|先頭|へ|新項目1|を|挟む|
|
|新項目2|という|リンク項目|を|作る|
|新項目2|の|値|を|「B」|とする|
|先頭|へ|新項目2|を|挟む|
|
|結果|は|、|先頭|から|「A」|を|探したもの|
|結果|の|値|を|表示する|
|
|リンク項目|とは|
    |次項目|を|持つ|
    |値|を|持つ|
|
    |［|値|］|を|、|探す|手順|
      |対象|を|自分|とする|
      |繰り返す|
        |もし|対象|の|値|が|値|なら|繰り返し|から|抜ける|
        |対象|を|対象|の|次項目|とする|
        |もし|対象|が|無参照|なら|繰り返し|から|抜ける|
      |繰り返し終わり|
      |対象|を|返す|
    |終わり|
|
    |［|自分|］|へ|、|［|項目|］|を|、|挟む|手順|
      |項目|の|次項目|は|、|自分|の|次項目|
      |自分|の|次項目|は|、|項目|
    |終わり|
|
    |［|自分|］|を|、|消す|手順|
      |項目|を|次項目|とする|
      |自分|の|次項目|を|項目|の|次項目|とする|
      |項目|の|次項目|を|無参照|とする|
    |終わり|
|終わり|
```

参考文献

[1] 筧研究室，“日本語プログラミング言語「プロデル」”，http://www.kake.info.waseda.ac.jp/research/，2009 年時点.

[2] 岡田健，中鉢欣秀，鈴木 弘，大岩元，“プログラミング言語としての日本語”，第 44 回プログラミングシンポジウム報告集 (2003).

[3] ゆうと，“日本語プログラミング言語「TTSneo」”，http://tts.utopiat.net/，2009 年時点.

[4] クジラ飛行机，“日本語プログラミング言語「なでしこ」”，http://nadesi.com/，2009 年時点.

[5] 中鉢欣秀，大岩元，“ Java ヴァーチャルマシンをターゲットとした日本語オブジェクト指向言語の開発”. 情報処理学会研究報告. PRO 97(45) pp.31-36 (1997).

[6] 三上章，“象は鼻が長い―日本文法入門”，くろしお出版 (1960).

[7] 山崎紀美子，“日本語基礎講座―三上文法入門”，ちくま新書 (2003).