6B-7

日本語プログラミング言語処理系の字句解析

馬場 祐人†

†早稲田大学理工学術院 基幹理工学研究科

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，日本語の語順でプログラムを書くプログラミング言語である．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，空白や読点で単語を分かち書きしないことが特徴である．一般に自然言語として日本語文を解析するには，形態素解析が用いられる．形態素解析では文法が考慮されず，コスト計算によって文を形態素に区切る．一方日本語プログラムの解析では，日本語プログラミング言語の文法規則に則ってプログラム文を字句に区切られることが求められる．

本発表では，日本語プログラミング言語処理系で分かち書きされない日本語プログラムを解析するために，字句解析と構文解析を同時に行ってプログラム解析する手法について説明する．また実用的なソフトウェア開発環境である日本語プログラミング言語“プロデル[1]”にこの手法を実装し，実践を通じて有用性を議論する．

2. 日本語プログラムの構文

日本語プログラミング言語は，普段読み書きする日本語文の特徴に倣って，空白や読点で字句を分かち書きせずに自然な文に近い書き方でプログラムを書ける．

図 1 は，Java 言語で書いた関数を呼び出すプログラムであり，図 2 は，それをプロデルの文法で書いたプログラムである．図 1 で示すように，C 言語や Java 言語などの一般的なプログラム言語での字句は，空白や記号で区切られる．一方，図 2 で示すようにプロデルの字句は，空白や記号で区切られない．

一般の字句解析器は， Lex や Flex といった既存のジェネレータで作ることができる．これらを使って日本語プログラムを解析する字句解析器を作るには，助詞に使う文字を字句の区切りにして切り出す方法を採る．この方法では識別子に助詞で使う文字が使えず，言語仕様に大きな制約が生まれる．本研究では，字句解析器に辞書を持たせ，辞書の登録語によって字句を切り出し，また構文解析と組み合わせてプログラムを解析する手法について有用性を検証した．

A Tokenizer For The Japanese Programming Language.
†BANBA Yuto, Graduate School of Fundamental Science and Engineering, Waseda University.

```
コピーする(″文章.txt″, 保存先);
```

図 1 Java 言語の関数呼出し文

```
「文章.txt」を保存先へコピーする。
```

図 2 プロデルの関数呼出し文

3. プロデルにおけるプログラム解析手法

3.1. 字句解析で使う名前木

字句解析では，数字や記号，空白，コメントを字句として切り出す他に，名前木にある登録語を使って字句を切り出す．名前木とは，コンパイラにおける名前表に相当する．名前木には登録語にその品詞(識別子の種類)を付加して格納する．登録語は，予約語，プログラム内で宣言された識別子，参照する外部ライブラリで宣言された識別子である．プロデルで使われる品詞は，名詞，助詞(助数詞を含む)，動詞，および記号(句読点，演算子，括弧など)である．名詞は，識別子(型，変数)である．なお，日本語プログラムには“もし”や“または”といった上記以外の品詞が含まれるが，これらの語は品詞として分類するだけの単語数が少ないため，まとめて予約語として扱う．

名前木のデータ構造は，トライ木である．名前木の節や葉はそれぞれスタック構造とする(図 3)．名前木にある登録語の中には，宣言されたスコープによって品詞が異なる語がある．スタック構造にして，字句を切り出す際にスタックの最上部の品詞を選択することで，スコープに応じて適切な品詞の字句を切り出せる．

3.2. 字句解析と同時に行う構文解析

本手法による日本語プログラムの解析では，文法規則に合う品詞を持つ字句を切り出しながら構文解析する．字句解析では，ある文字位置で複数の字句を取り得る場合があり，この際には，文法規則に合う品詞の字句を切り出す．

例えば，“整数として 10 を代入する”というプログラム文を解析するとする．なおプロデルの関数呼出し文には，関数呼出し文::={式 助詞}動詞という文法規則がある．図 4 のように 2 単語目は，“として”と“と”のどちらかの字

句を取り得る．“と”を切り出すと，次の字句も助詞となり文法規則に合わない．結果として文法規則に合う“として”を切り出す．

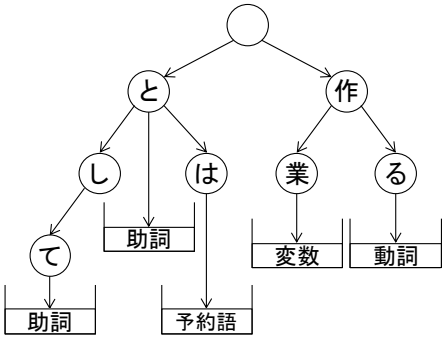

図 3 名前木の構造

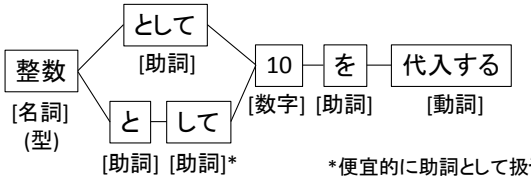

図 4 構文解析と字句解析を組合せた字句の切り出し

3.3. プログラム内で宣言される識別子

解析対象のプログラムの中ではじめて宣言される識別子は，字句解析を始めた時点では名前木に登録されていない．そのため 1 パスでは，正確に字句を切り出せない．そこでプログラム解析中に名前木にない語が表れた際は，ひとまずその語を未知語として扱って構文解析する．解析後，新たに定義された識別子を名前木に登録する．そして再度，その名前木を使って字句解析と構文解析を行い，プログラムを解析する．

4. 評価

4.1. 日本語プログラムの解析への適用

本手法を実装したプロデルのプログラム解析器を使い，プロデルで作成した複数のプログラム(図 6)が正しく解析できるかどうかを検証した．

4.2. 字句分割の曖昧さ

日本語プログラムでは，複合語を一つの識別子として扱う．そのため同じ品詞が連続する区切り方はなく，複合語によって字句の区切り方が一意に決まらない区切りの曖昧さは起こらない．またプログラムで使用される字句は，プログラムで宣言される語か，外部で宣言された語であるため未知語は現れない．そのため構文解析で明確に区切り方が定まるため，未知語による区切りの曖昧さも起こらない．

例えば“お気に入りリストへ URL を加える”というプログラム文を字句で区切ることを考える．このとき，名前木には{お気に入りリスト(名詞)，お気に入り(名詞)，リスト(名詞)，URL(名詞)，に(助詞)，へ(助詞)，を(助詞)，加える(動詞)}が登録されているとすると，図 5 に示した (A)～(D)の区切り方が考えられる．

お気 | に | 入り | リスト | へ | URL | を | 加える	…(A)
お気 | に | 入りリスト | へ | URL | を | 加える	…(B)
お気に入り | リスト | へ | URL | を | 加える	…(C)
お気に入りリスト | へ | URL | を | 加える	…(D)

図 5 考えられる字句の区切り方

提案手法では未知語が現れないため，(A)，(B)のようには区切られない．(C)は，名詞が連続するので構文として誤りである．(D)だけが文法として正しい区切り方である．

仮に，名前木に{お気(名詞)，入りリスト(名詞)}が含まれるとすると，(B)と(D)のどちらも文法として正しい区切り方となる．この場合，動詞“加える”へ助詞“へ(に)”と“を”の 2 つが係ることが宣言されていれば，意味解析の時点で誤りとなる．本手法を適用したプロデルで実際にプログラムを作成する際には，このような特異な識別子の名付け方をしなければ，字句の区切りが一意に定まった．

字句解析と構文解析を同時に行うことで，字句解析だけでは区切り方が一意に決まらない場合でも日本語プログラムの文法知識を用いて適切に字句を切り分けられる．

5. まとめ

本研究では，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プログラムの解析を言語処理系で実現するために，名前木を構築し，字句解析と構文解析を同時に行う手法を考案した．本手法を実際の処理系に実装し，日本語プログラムを問題なく解析できた．

プログラム	有効文字数	有効字句数	有効行数
クラス図生成ツール	39,767	15,141	2,189
ブロック崩しゲーム	27,535	12,328	1,888
書店販売管理Webシステム	33,314	11,018	1,675
Javaバイトコード解析ツール	26,082	11,038	1,386
C言語風コンパイラ	15,778	5,965	1,127
ブログシステム	15,967	5,678	864
シューティングゲーム	8,160	3,151	692
ホテル予約システム	8,419	2,912	559
BASICコンパイラ	5,613	1,766	456
wikiシステム	6,498	2,408	418
IRCクライアント	3,482	1,345	233
合計	190,615	72,750	11,487

図 6 検証した日本語プログラムの規模

参考文献

[1] “日本語プログラミング言語「プロデル」”，http://rdr.utopiat.net/，2013 年時点．