A-013

形態素解析エンジンを利用した日本語プログラミング言語 Wowrus の開発

Development of a Japanese programming language "Wowrus" by utilizing Morphological Analysis

利根川　翔†　　筧　捷彦‡

Sho TONEGAWA　Katsuhiko KAKEHI

1. はじめに

日本語プログラミング言語とは，予約語や識別子を日本語で記述し，また助詞を使い日本語の語順に近い文法で記述するプログラミング言語である [1]．プログラミングに母国語を用いることによりプログラムの可読性が向上することが確認されており [2]，日本語プログラミング言語は日本人にとって有用であると言える．

近年の日本語プログラミング言語の多くはプログラムの可読性の向上を図っているが，分かち書きが必要な場面が残っているほか，記号が多用されており理解しづらいなど課題がある．我々は，これらの課題を解決する方法として，字句解析や構文解析に形態素解析エンジンを利用する手法を提案する．本稿では，記号を削減し，分かち書きに頼ることの少ない構文を実現する日本語プログラミング言語を，形態素解析エンジンを利用して開発し，形態素解析を用いることの長所・短所を報告する．

2. 既存の日本語プログラミング言語の問題点

近年は，自然な日本語表現でプログラムを書けるようにすることを目的とした言語が多く開発されている．最近の日本語プログラミング言語の例に，プロデル [1]，クロガネ，なでしこ [3]，言霊 [4]などがある．これらの言語は，自然言語の日本語とほぼ同様の語順でプログラムが書けるなど，可読性の向上が図られている．

しかし，これらの言語でも，分かち書きや特殊な記号によって可読性が損なわれてしまう問題が残っている．例えば，プロデルで手順(関数)を定義する際には，引数を【 】と読点で分かち書きしなければならない．一方，なでしこは“●”が関数定義を表すなど記号を多用しているが，記号の現れる文を理解するには各言語における定義を知らなければならず，プログラムの可読性は下がってしまう．

3. 提案手法

2 章で述べた問題点を改善する方法として，字句解析や構文解析に形態素解析エンジンを利用する手法を提案する．形態素解析とは，与えられた文を形態素単位に区切る自然言語処理の基礎技術である [5]．

プログラムを形態素解析することで，形態素と，それに付属する品詞情報が得られる．品詞情報を構文解析に利用することで，分かち書きに頼ることの少ない解析が可能になり，使用する記号を削減できると考えた．

4. 日本語プログラミング言語 Wowrus の開発

3 章で示した手法の実装例として，日本語プログラミング言語 Wowrus(わおるす)の開発を行った．形態素解析エンジンは Sen [6]を利用した．

4.1. Wowrus プログラム例

数値 N を引数に，N 番目のフィボナッチ数を返す関数を，Wowrus で記述した例をリスト 4.1 に示す．

```
値をフィボナッチするとは…
　　まず値≦1 であるなら…値を返す。
　　最後に、(値-1 をフィボナッチしたもの)
　　+(値-2 をフィボナッチしたもの)を返す。
```

リスト 4.1　Wowrus のプログラム例(フィボナッチ数)

4.2. 字句解析・構文解析手順

リスト 4.1 の一部(リスト 4.2)を例に，Wowrus の言語処理系がプログラムを解析する手順を説明する．

```
値≦1 であるなら…値を返す。
```

リスト 4.2　解析例に用いるプログラム

4.2.1. 形態素解析

従来のプログラミング言語の字句解析にあたる処理を，Sen による形態素解析によって行う．Wowrus のプログラムを形態素解析した結果の例をリスト 4.3 に示す．

プログラムを解析する際には，「」や" "で囲んだ部分を 1 つの名詞として扱う等の特殊な対応が必要であり，そのための処置を Wowrus 言語処理系が先に行う．その後，Sen がプログラムを形態素に区切り，各形態素に品詞情報を付加する．リスト 4.3 では，品詞情報を【 】内に記す．

```
+ 値（値）　　【名詞-一般】
+ ≦（≦）　　【未知語】
+ 1（1）　　【名詞-数】
+ で（で）　　【助詞-格助詞-一般】
+ ある（ある）　【動詞-自立】
+ なら（だ）　　【助動詞】
+ …（…）　　【記号-一般】
+ 値（値）　　【名詞-一般】
+ を（を）　　【助詞-格助詞-一般】
+ 返す（返す）　【動詞-自立】
+ 。（。）　　【記号-句点】
```

リスト 4.3　形態素解析結果の例

4.2.2. 文節生成

各形態素は，持っている品詞情報から自立語と付属語に分類される．Wowrus 言語処理系は，自立語に，それに続く付属語をつなげて文節を生成し，それを構文解析時の最小単位とする．文節生成結果の例をリスト 4.4 に示す．

生成された文節は，自立語や付属語の品詞情報によって分類される．リスト 4.4 では，文節分類を≪≫内に記す．

```
+ 値　　≪名詞単独≫
+ ≦　　≪記号≫
+ 1で　　≪補語≫
+ あるなら　≪述語≫
+ …　　≪三点≫
+ 値を　　≪目的語≫
+ 返す　　≪述語≫
+ 。　　≪句点≫
```

リスト 4.4　文節生成結果の例

4.2.3. 構文解析

生成された文節の分類を利用して構文解析を行い，構文木を生成する．構文解析結果の例をリスト 4.5 に示す．

<述語>が関数呼出し，<文要素>が引数の役割を果たす．

† 早稲田大学大学院基幹理工学研究科情報理工学専攻
Department of Computer Science and Engineering, Graduate School of Fundamental Science and Engineering, Waseda University.
‡ 早稲田大学理工学術院
Faculty of Science and Engineering, Waseda University.

```
+〈条件付文〉
|  +〈文〉
|  |  +〈文要素〉
|  |  |  +〈論理式（補語)〉
|  |  |  |  +〈名詞単独〉
|  |  |  |  |  + 値
|  |  |  |  + ≦
|  |  |  |  +〈数値（補語)〉1.0 で
|  |  +〈述語〉
|  |  |  + あるなら
|  + …
|  +〈文〉
|  |  +〈文要素〉
|  |  |  +〈目的語〉
|  |  |  |  + 値を
|  |  +〈述語〉
|  |  |  + 返す
```

リスト 4.5　構文解析結果の例

4.3.　機能と実績

Wowrus は，オブジェクト指向にはまだ対応していないが，基本的な制御構文や数値，真偽値，文字列，配列などを扱うことができる．プログラム例として数値演算，配列の処理，文字列処理などを行うプログラムを実装し，動作することを確認した．現在，オブジェクト指向への対応に向けて追加実装中である．

Wowrus は，以下の Web ページにて公開中である．
http://www.kake.info.waseda.ac.jp/research/wowrus.html

5.　形態素解析を用いることの長所

Wowrus は，形態素解析によって得られた品詞情報を用いた解析を行うため，記号による分かち書きを必要とする場面が減り，使用する記号の種類を削減した可読性の高い構文を実現する効果が見込まれる．

5.1.　使用する記号の比較

プロデル，なでしこ，Wowrus について，プログラム中で使用される記号がそれぞれ何種類あるか調べ，その数を比較した．なでしこで使用される記号は公式 Web サイト [3]のマニュアルの「基本文法」関連のページを参考に，プロデルで使用される記号はプロデルの言語仕様書を参考に調査した．なお，Wowrus で未実装かつ他言語で実装されている機能がある場合，その機能に関する記号は数に含めない．

結果を表 5.1 に示す．この結果より，他の言語と比べ，Wowrus は記号の少ない構文を実現したことが分かる．

	記号の種類
なでしこ	16
プロデル	13
Wowrus	9

表 5.1 プログラム中に現れる記号の種類

5.2.　対応助詞の数および使用時の分かち書きの比較

プロデル，なでしこ，Wowrus について，関数(プロデルの場合は手順)の定義・呼出し時に使用できる助詞が何種類あるか，また，その際に分かち書きが必要か否かを調べ，比較した．結果を以下にまとめる．

- なでしこ…27 種，関数定義時に語順が逆転，( ) による分かち書きが必要
- プロデル…助詞をユーザが定義可能，手順定義時に【 】と読点による分かち書きが必要
- Wowrus…助詞 228 種，助数詞 564 種を助詞として使用可能，関数定義・呼び出し時に分かち書き不要

以上より Wowrus は，なでしこの 30 倍に近い数の単語を助詞として用いることができることがわかった．プロデルに自由度では劣るものの，分かち書きに頼らずに多様な助詞を使うことが可能となっていることが分かる．

5.1 節の結果とあわせて，プログラム文の解析に形態素解析を用い，品詞情報を利用することは，プログラム文中で分かち書きを必要とする場面を減らし，使用する記号の種類を削減する効果があることがわかった．

6.　形態素解析を用いることの短所

6.1.　未知語への対応の必要性

形態素解析エンジンは辞書に含まれない単語に対応できないため，Sen の辞書にない単語を使用する場合，" " で囲むことで 1 つの名詞として扱うという特殊な対応が必要になる．未知語を使いたいときには記号を多用することになるほか，使用した単語が未知語か否かは解析するまで判らず，プログラムを書く際に不便になってしまう．

形態素解析エンジンと連携した開発環境を用意すれば，入力時に未知語か否かを確認することが可能になるため，この短所を軽減できる．

6.2.　Sen の解析能力への依存

Sen は自然言語処理用の形態素解析エンジンであり，プログラムの解析に特化しているわけではない．テスト中に“1+1+1+1+1+1+1”を Sen に解析させようとしたところ，未知語として読まれてしまう問題が見つかった(解決済)．このような問題は他にも多くあると予想され，それらを発見し，解決するのは非常に困難である．Sen の辞書データを編集するなど，根本的な解決が必要と考えられる．

7.　まとめ

形態素解析エンジンの日本語プログラミング言語解析への利用は，分かち書きを必要とする場面を減らし，使用する記号の種類の削減を実現することで，可読性の向上につながることが分かった．一方で，未知語への対応，形態素解析エンジンの解析能力への依存という短所が見つかった．特に，形態素解析エンジンの解析能力への依存は大きい．これを解決できれば，形態素解析エンジンの利用は，日本語プログラミング言語の可読性を高める上で非常に有用であろう．

参考文献

1. **ゆうと.** 日本語プログラミング言語「プロデル」. (オンライン) (引用日: 2011 年 1 月 11 日.) http://rdr.utopiat.net/.
2. **中川正樹, ほか.** 母語プログラミングの理念,実現,実践とその効果. 電子情報通信学会論文誌 Vol.J77-D-1, No.5(19940525) pp. 364-374. 1994.
3. **クジラ飛行机.** 日本語プログラミング言語「なでしこ」について. (オンライン) (引用日: 2010 年 12 月 24 日.) http://nadesi.com/about-nadesiko/.
4. **岡田健, ほか.** 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　2003.1, p.63-72. 2003.
5. **山下達雄.** 日本語形態素解析入門 Version.0.91, p.2. (オンライン) 1998 年. (引用日: 2010 年 12 月 22 日.) http://nais.to/~yto/doc/tech/jma/jma19990514.pdf.
6. **SenProject.** Sen Project. (オンライン) (引用日: 2011 年 1 月 11 日.) http://ultimania.org/sen/.